Irina Munteanu, Daniel Ioan
Universitatea "Politehnica" din Bucureşti
Laboratorul de Metode Numerice

Utilizarea Instrumentelor Unix în Ştiinţă şi Inginerie

1998

Cuprins

Introducere **9**

1 Utilizarea sistemului de operare Unix **17**

1.1 Intrarea în sistem 21

1.2 Lucrul cu ferestre XWindow 21

1.3 Ieşirea din sistem 24

1.4 Sesiunea de lucru şi configurarea ei 24

1.5 Interpretorul de comenzi 26

1.6 Comenzi pentru lucrul cu fişiere şi directoare 26

1.6.1 Structura de directoare şi fişiere 26

1.6.2 Comenzi pentru lucrul cu fişiere 29

1.6.3 Comenzi pentru lucrul cu directoare 30

1.6.4 Schimbarea drepturilor asupra fişierelor/directoarelor 30

1.6.5 Căutări 31

1.6.6 Comparări de fişiere 31

1.7 Indirectări 32

1.8 Imprimarea fişierelor 32

1.9 Lansarea şi oprirea comenzilor 33

1.10 Comenzi pentru controlul proceselor şi utilizatorilor 33

1.11 Ajutor! 34

1.12 Transferul fişierelor între DOS şi Unix 34

1.13 Diverse utilitare 34

1.14 Exerciţii 35

2 Editarea textelor — vi şi Emacs — **39**

2.1 Editarea textelor cu vi 40

2.1.1 Apelare 40

2.1.2 Moduri de lucru 41

2.1.3 Modul comandă 42

2.1.4 Setare opţiuni 45

2.1.5 Câteva facilităţi ale interfeţei grafice cu utilizatorul 45

2.2 Editarea textelor cu emacs 47

2.2.1 Lansare şi ieşire din Emacs 47

2.2.2 Ecranul Emacs 47

2.2.3 Comenzi de deplasare a cursorului 49

2.2.4 Comenzi de ştergere 49

2.2.5 Comenzi de formatare–indentare 49

2.2.6 Comenzi pentru blocuri 50

2.2.7 Comenzi pentru fişiere (încărcare-salvare) 50

2.2.8 Căutare şi înlocuire 50

2.2.9 Comenzi pentru ferestre 50

2.2.10 Lucrul cu macrocomenzi 50

2.2.11 Comenzi pentru dezvoltarea programelor 51

2.2.12 Diverse 51

2.3 Exerciţii 52

3 Poşta electronică — elm — **55**

Conceptele specifice poştei electronice 57

3.1 Apelare 59

3.2 Structura ecranului elm 59

3.3 Citirea mesajelor primite 59

3.4 Trimitere de mesaje 60

3.5 Pseudonime (aliasuri) 61

3.6 Ştergerea / mutarea mesajelor în altă căsuţă poştală 62
3.7 Modificarea opţiunilor 62
3.8 Ieşirea din `elm` 64
3.9 Exerciţii 64

4 Editarea documentelor — LATEX — **67**

4.1 Etapele utilizării LATEX 70
4.2 Structura unui fişier LATEX 70
4.3 Comenzi LATEX 74
4.4 Moduri de lucru LATEX 75
4.4.1 Modul paragraf 75
4.4.2 Modul matematic 77
4.4.3 Modul LR 78
4.5 Cadru LATEX 79
4.6 Macrodefiniţii LATEX 80
4.7 Aspecte speciale ale utilizării LATEX 81
4.7.1 Caractere speciale 81
4.7.2 Accente 81
4.7.3 Despărţirea în silabe 82
4.8 Structura unei pagini 83
4.9 Fişiere rezultate în urma procesării cu LATEX 83
4.10 Câteva exemple 83
4.10.1 Fişierul `latex_head.tex` 83
4.10.2 Un exemplu de document 85
4.10.3 Un exemplu de articol în stilul IEEE Trans. on Magnetics 92
4.11 Gestionarea referinţelor bibliografice folosind BIBTEX 96
4.11.1 Structura unui fişier BIBTEX 97
4.11.2 Introducerea referinţelor bibliografice în fişierele LATEX 99
4.11.3 Fişiere utilizate în BIBTEX 100
4.11.4 Exemple de utilizare 101

4.12 Configurarea LaTeX de către utilizator 104
4.13 Realizarea planşelor de prezentare folosind stilul `seminar` 105
4.13.1 Comenzi specifice stilului `seminar` 106
4.13.2 Exemplu de utilizare 107
4.14 Folosirea programului utilitar `make` 109
4.15 Exerciţii 110

5 Grafice şi figuri — Gnuplot, XFig, XPaint, XV — 121

5.1 XFig 125
5.1.1 Apelare 125
5.1.2 Obiecte XFig 125
5.1.3 Regiuni ale ecranului 126
5.1.4 Salvarea desenului în alte formate 128
5.2 Gnuplot 129
5.2.1 Apelare 129
5.2.2 Comenzi utile pentru început 129
5.2.3 Manualul pentru `gnuplot` 129
5.2.4 Ce poate face `gnuplot` 129
5.2.5 Comenzi `gnuplot` 130
5.2.6 Comenzi de bază: `plot, splot` 130
5.2.7 Alte comenzi utile 133
5.2.8 Salvarea graficelor în alte formate 135
5.2.9 Fişiere de comenzi `gnuplot` 136
5.3 XPaint 138
5.3.1 Apelare 138
5.3.2 Ferestre `xpaint` 138
5.3.3 Fereastra de instrumente 138
5.3.4 Fereastra de desenare 139
5.3.5 Salvarea desenului în diferite formate 140

5.4 XV 142
5.4.1 Apelare 142
5.4.2 Ferestre `xv` 142
5.4.3 Operaţiuni în fereastra de imagine 142
5.4.4 Operaţiuni în fereastra de control 143
5.5 Exerciţii 145

6 Analiza numerică a modelelor matematice — Scilab — 147

6.1 Apelare Scilab 150
6.2 Componente Scilab 150
6.3 Interfaţa grafică a programului Scilab 150
6.4 Câteva noţiuni introductive 151
6.4.1 Formatul comenzilor Scilab 151
6.4.2 Facilităţi 152
6.4.3 Demonstraţie 152
6.5 Tipuri de date Scilab 152
6.5.1 Constante 152
6.5.2 Matrice de scalari 153
6.5.3 Matrice de şiruri de caractere 154
6.5.4 Matrice de polinoame şi de funcţii raţionale 155
6.5.5 Matrice cu elemente de tip logic 155
6.5.6 Liste 156
6.6 Programarea în Scilab 156
6.6.1 Funcţii Scilab 156
6.6.2 Funcţii predefinite în Scilab 157
6.6.3 Tipuri de instrucţiuni Scilab 157
6.7 Fişiere de comenzi 158
6.8 Grafice 159
6.9 Exemple 160
6.10 Apelul unei rutine C sau FORTRAN din Scilab 162
6.11 Traducerea automată în FORTRAN a unei funcţii Scilab 165
6.12 Exerciţii 166

7 Medii de programare Unix — C, C++, FORTRAN — 169

7.1 Principalele opţiuni ale compilatoarelor `gcc, g++, g77` 172
7.2 Recomandări privind stilul de lucru profesional 173
7.3 Compilarea programelor folosind `make` 174
7.3.1 Structura unui fişier `makefile` 174
7.3.2 Modele de fişiere `makefile` 175
7.4 Utilizarea depanatorului `gdb` 176
7.5 Apelarea rutinelor FORTRAN dintr-un program scris în limbajul C . . . 179
7.6 Exerciţii 182

8 Comunicaţia între sistemele Unix — Internet — 193

8.1 `telnet` 200
8.2 `write` 200
8.3 `talk` 201
8.4 `ftp` 203
8.5 Alte comenzi utile, legate de transferul de fişiere şi de comunicaţie 204
8.6 Elemente de bază WWW 205
8.7 Structura limbajului HTML 207
8.8 Coduri de formatare HTML 208
8.8.1 Principalele perechi de coduri de formatare HTML 208
8.9 Un exemplu de document HTML 210
8.10 Exerciţii 212

ANEXE 215

A Teste finale de laborator 215

B Regulamentul Laboratorului de Metode Numerice 221

C Evoluţia Laboratorului de Metode Numerice – LMN 223

D Resursele Laboratorului de Metode Numerice 229

D.1 Resurse hardware 230

D.2 Resurse software 237

D.3 Interfaţa grafică a butoanelor mouse-ului 239

D.3.1 Meniuri de comenzi disponibile sub Linux 239

D.3.2 Meniuri de comenzi disponibile în reţeaua de staţii grafice 242

Bibliografie şi Webografie 247

Index 250

Introducere

Motto: "Cercetarea ştiinţifică înseamnă 1% inspiraţie şi 99% folosirea eficientă a unor instrumente adaptate".

De mai bine de două decenii, la clasicele "unelte de lucru" ale cercetătorului (teoria şi experimentul) s-a mai adăugat unul: calculatorul. Viziunea modernă asupra cercetării, impusă nu fără obiecţii şi dificultăţi, plasează metodele domeniului numit în SUA "*Computational Science and Engineering* – CSE" [54] pe acelaşi nivel cu abordările teoretice şi cele experimentale (fig. I.1).

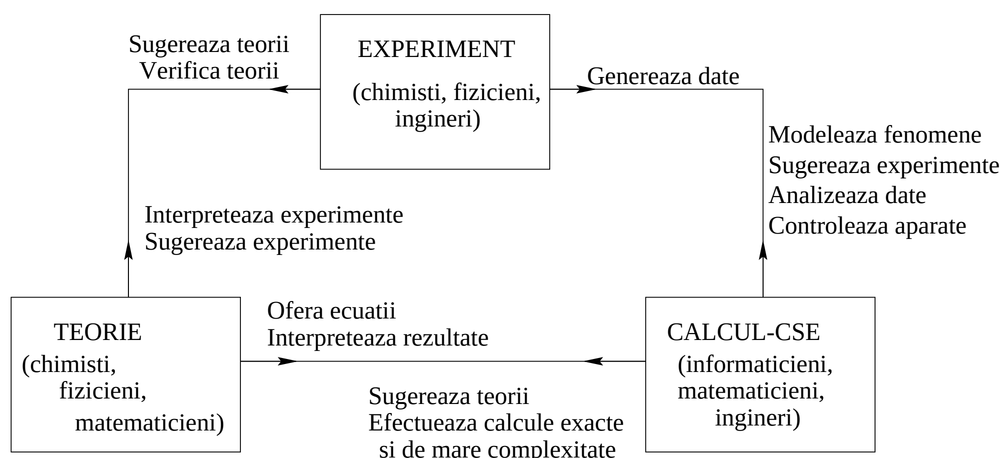


Fig. I.1: Poziţia Ştiinţei şi Ingineriei Computaţionale (CSE) în tabloul ştiinţei

Dar lucrurile nu se opresc aici: cercetarea nu este importantă în sine, ci prin impactul ei asupra societăţii. Metoda uzuală de valorificare a rezultatelor cercetării este cea de diseminare a cunoştinţelor descoperite, prin publicare. Prezenţa calculatoarelor în orice laborator ştiinţific a determinat o mutaţie şi în acest domeniu: cercetătorul modern a devenit, practic, propriul său tehnoredactor şi tipograf!

Mai mult, mediile de comunicare profesională migrează de la tiparul clasic pe hârtie spre formatul electronic, cu toate implicaţiile acestuia: formele de reprezentare şi de transmitere a cunoştinţelor ştiinţifice se diversifică, migrând de la documentul scris alcătuit în cel mai complicat caz din text, tabele, formule matematice şi chimice sau figuri, spre

"documente" electronice ce conţin în plus sunete, imagini 3D, animaţii sau secvenţe video; interactivitatea începe să fie incorporată în structura intimă a producţiei, de la simpla structurare hipertextuală (mai corect *hipermedia*) până la interactivitate completă şi puternică.

Potenţate de dezvoltarea spectaculoasă a tehnologiei informaţiei, CSE şi noua filozofie a comunicării profesionale vor avea, cu certitudine, efecte sociale majore în secolul următor.

Modul în care ar trebui predat noul domeniu ştiinţific —CSE— studenţilor din universităţile americane a stârnit o extrem de interesantă discuţie în câteva numere succesive ale *IEEE Computational Science and Engineering* (numerele 2-3 din 1996). Părerile exprimate au fost extrem de diverse, începând cu afirmaţia că totul ar trebui să constea de fapt în însuşirea instrumentelor Unix, şi sfârşind cu opinia că stăpânirea mediului *Mathematica* este suficientă şi că specialiştii în calculatoare mai mult încurcă decât ajută la formarea inginerilor în utilizarea tehnologiilor CSE. Prezenta lucrare poate fi considerată ca o replică a Laboratorului de Metode Numerice din Politehnica Bucureşteană, în această dispută.

Revenind la motto-ul de la începutul introducerii, manualul de faţă nu se referă la acel procent de inspiraţie necesar cercetării creatoare (care de altfel nu poate fi învăţată) şi nici măcar la conţinutul propriu-zis al cercetării, ci la **atelierul de lucru** al cercetătorului modern.

Firul conducător al acestui manual este de a descrie, într-o manieră progresivă care să permită înţelegerea şi învăţarea uşoară, folosirea instrumentelor necesare redactării documentelor ştiinţifice, în special ale celor din domeniul ştiinţei şi ingineriei computaţionale.

Iniţial, prezenta lucrare a fost concepută ca un îndrumar de laborator pentru studenţii anului VI (studii aprofundate) de la Facultatea de Electrotehnică din Universitatea "Politehnica" din Bucureşti la direcţiile "Proiectarea Asistată de Calculator a Dispozitivelor Electromagnetice" şi "Proiectarea microsistemelor". Aceşti studenţi îşi desfăşoară activitatea în Laboratorul de Metode Numerice şi a fost nevoie de o instruire specială în folosirea mediului hardware/software avansat care este disponibil în acest laborator. Primele materiale conţinute în acest manual au fost elaborate începând din 1995 de Irina Munteanu şi au fost puse la dispoziţia studenţilor, în format electronic, în pagina de Web a laboratorului (`http://www.lmn.pub.ro`).

O analiză mai atentă ne-a făcut să înţelegem că experienţa acumulată, mai ales în ultimii 5 ani, merită să fie distribuită într-un cadru mult mai larg. În consecinţă, am elaborat acest **manual, destinat tuturor celor interesaţi în folosirea profesională a calculatoarelor în activitatea ştiinţifică şi inginerească** (studenţi, ingineri, cercetători în orice domeniu ştiinţific). Deşi manualul nu se adresează în mod explicit studenţilor sau specialiştilor în calculatoare, experienţa din laboratorul nostru, în care lucrează şi astfel de persoane, ne-a demonstrat că o asemenea lucrare le poate fi extrem de utilă.

Această lucrare este o **premieră naţională**, din mai multe puncte de vedere:

- în primul rând este un **bun început** pentru oricine are acces la un calculator personal cu un sistem de operare de tip Unix şi aspiră să-l folosească pentru a obţine rezultate profesionale de înalt nivel, folosind tehnicile cele mai avansate;
- lucrarea nu înlocuieşte manualele de referinţă (dar **sintetizează** câţiva "metri" de raft de astfel de manuale, citate la bibliografie), pe care va trebui să le studiaţi dacă vreţi să vă specializaţi în utilizarea unuia din instrumentele prezentate;
- lucrarea nu este un memorator sistematic şi exhaustiv (gen ". . . *in a nut shell*") ci este un **compendiu selectiv**, un memorator în care sunt descrise pe scurt comenzile care s-au dovedit a fi cele mai utilizate în activitatea curentă din laboratorul nostru;
- instrumentele Unix descrise alcătuiesc un **sistem coerent şi complet** ("*toolbox*") pentru realizarea unor cercetări ştiinţifice la nivel mondial, precum şi pentru valorificarea rezultatelor obţinute, prin publicarea şi prezentarea lor, conform celor mai exigente standarde actuale;
- instrumentele prezentate fac parte din categoria **software cu licenţe gratuite**, o categorie extrem de amplă şi în dezvoltare, dar din care selecţia pachetelor profesionale, cu performanţe peste nivelul produselor comerciale, nu este simplă;
- exemplele date în lucrare au fost testate, exerciţiile au fost parcurse mai mulţi ani la rând de studenţi şi **îmbunătăţite în urma experienţei câştigate** iar multe dintre "trucurile" prezentate au necesitat zile de muncă pentru a fi descoperite;
- manualul a fost realizat în întregime **folosind chiar instrumentele prezentate**, ceea ce validează încă o dată puterea mediului propus.

Citirea atentă a introducerilor, care conţin **descrierea** principalelor concepte (însoţite de trimiteri la bibliografie), completată cu parcurgerea conştiincioasă a exerciţiilor reprezintă o cale posibilă ("calea regală") de a **înţelege repede şi corect** domeniul extins pe care acest manual încearcă să-l acopere.

Decizia curajoasă de a prefera calculatoare controlate de sisteme de operare Unix, aflat pe locul al doilea din punctul de vedere al numărului de utilizatori, a fost determinată de constatarea că sistemul MS-Windows deţine supremaţia de vânzări mai ales datorită aplicaţiilor casnice, jocuri şi birotică şi mai puţin prin aplicaţiile lui profesional-ştiinţifice. Deoarece numele UNIX (cu majuscule) este marcă înregistrată, în acest manual se preferă utilizarea numelui Unix (cu litere mici!) pentru sistemele de operare *de tip* UNIX.

De fapt, Unix este mai mult decât o familie de sisteme de operare, este o **cultură** a unei întregi comunităţi care foloseşte reţeaua de calculatoare Internet şi care adoptă un stil specific de programare şi de rezolvare a problemelor, prin combinarea în structuri elegante a problemelor simple. Poate cea mai bună definiţie a acestui concept este cea dată de Hahn [55]: "*Unix is a set of tools for smart people*"[1].

[1] *Unix este un set de instrumente pentru oameni inteligenţi.*

Contribuţia autorilor la lucrare este următoarea:

I. Munteanu (80%): Descrierea programelor şi comenzilor (cu excepţia Emacs), conceperea majorităţii exerciţiilor, tehnoredactare, corectură (80%);

D. Ioan (20%): Introducerile capitolelor, introducerea lucrării, descrierea Emacs, anexele referitoare la istoria şi la resursele Laboratorului de Metode Numerice, corectură.

Mulţumiri

Apariţia acestui manual nu ar fi fost posibilă fără înfiinţarea Laboratorului de Metode Numerice (LMN), dezvoltat în cadrul Catedrei de Electrotehnică în Politehnica bucureşteană. Faptul că această catedră este una din cele mai prestigioase din universitate se datorează tradiţiei ei şi mai ales regretatului Academician *Remus Răduleţ*, care a fost personalitate proeminentă şi a cărui amprentă a rămas puternic imprimată în Catedră. Urmaşii acestuia, profesorii *C.I. Mocanu*, *Alexandru Timotin* şi *Andrei Ţugulea*, ne-au învăţat să muncim serios, să gândim riguros, dar şi cum să împărtăşim studenţilor cunoştinţele acumulate.

Pentru înfiinţarea LMN se cuvine să mulţumim în primul rând D-lui prof. *Constantin Nemoianu*, care a renunţat de bunăvoie la camera EB-206 unde a fost instalat pentru prima dată acest laborator. În perioada timpurie a LMN, un rol important l-au avut domnii *Alexandru Morega* (în prezent profesor la Catedra de Maşini Electrice din U.P.B.), *Costin Stancu* (în prezent "senior designer" la General Motor în California, USA) precum şi *Mihai Georgian* (în prezent şeful unei echipe de programatori în Canada), toţi trei absolvenţi ai seriei 1980 ai Facultăţii de Electrotehnică.

Tot în acea perioadă ne-au sprijinit: *Nini Popovici,* inginer de sistem la în Centrul de Calcul al Politehnicii (în prezent Directorul *RoEduNet*), cu ajutorul căruia am instalat primele terminale la distanţă (în LMN) ale sistemului FELIX C256; *Vlad Ţepelea,* şeful unui colectiv din I.T.C., beneficiarul pachetului LOCAP (în prezent preşedintele Consiliului de Administraţie RomTelecom); *Al. Nica,* directorul centrului de calcul CEPECA şi care ne-a împrumutat modem-urile pentru conexiunea WANG, acordate cu mare îndemânare de dl. *Sorin Popescu,* şef de lucrări la Facultatea de Electronică; *Sorin Gheţan,* administratorul sistemului WANG; d-na *Julieta Florea,* care în calitate de Decan al Facultăţii de Electrotehnică şi Energetică a acceptat în 1989 ca disciplina de Metode Numerice să fie predată de ingineri şi nu de matematicieni; *Sven Kratochvil,* talentatul programator al pachetului FAP în prima sa versiune, dar şi *Valentin Cotârţă,* cu care în 21 decembrie 1989 ne strecuram printre barajele "antitero" pentru a aduce în Politehnică primul calculator compatibil IBM PC-AT, importat după un adevărat "război" cu autorităţile de atunci ale Ministerului Învăţământului.

Mulţumiri se cuvin adresate şi celor peste 50 de prestigioşi specialişti din întreaga lume care ne-au răspuns la scrisorile de intenţie privind colaborarea interuniversitară scrise încă din ultima săptămână a anului 1989. O bună parte dintre aceştia au devenit partenerii şi prietenii noştri în cadrul proiectului *TEMPUS JEP 2717,* iniţiat în 1990 şi finanţat de *Comisia Europeană* prin *European Training Foundation* (ETF). Vizita la Politehnica din Torino, făcută ca urmare a invitaţiei de a-l însoţi pe profesorul Andrei Ţugulea (pe atunci membru în Guvernul României), a constituit pasul decisiv în pregătirea unui astfel de proiect. Aici am întâlnit parteneri desăvârşiţi ca prof. *Vicenzo Pozzolo,* preşedintele COREP, Rectorul *Rodolfo Zich, Aurelia Stepanescu* şi *Michele Patrissi.* Aceştia, împreună cu *Alain Bossavit* (Electricité de France), *Jürgen Arnold* (MSC MacNeal-Schwendler, München), *Stavros Christodoulakis* (Technical University of Crete), *Dominic Collard* (ISEN, Lille), *Bernard Courtois* (Institut National Polytechnique de Grenoble), *Peter Kayes* (Thames Valley University), *Giorgio Molinari* (Uni-

versitá di Genova), *Adel Razek* (Laboratoire de Génie Électrique de Paris), *Kurt Richter* (Technische Universität Graz), *David Rodger* (University of Bath), *Guglielmo Rubinacci* (Universitá di Cassino), *Jean-Claude Sabonnadière* (Institut National Polytechnique de Grenoble), *H. M. Schaedel* (Fachhochschule Köln), *John Tegopoulos* (National Technical University of Athens), au contribuit decisiv la succesul proiectului. Totuşi, acest lucru n-ar fi fost posibil fără contribuţia prof. *F.M.G. Tomescu,* membru în Comitetul Executiv al proiectului (în prezent director de studii aprofundate în catedra noastră), care a înţeles mai devreme decât alţi colegi de catedră necesitatea extinderii laboratorului.

Importul primelor staţii grafice cu procesor RISC nu a fost facil, datorită restricţiilor de export a tehnologiei strategice, iar pentru succesul acestei întreprinderi trebuie să le mulţumim reprezentanţilor producătorilor de staţii grafice: *Mircea Zaharescu* (firma Hewlett Packard), *Bogdan Cocora* (firma DEC), *Trandafir Moisa* (firma Sun) şi *Octavian Ţuţurea* (firma SGI). Poate mai dificilă decât cumpărarea echipamentelor a fost configurarea şi administrarea reţelei hibride în continuă creştere. Administratorii de reţea: *Mihai Lăzărescu* (în prezent în Italia), *Mihai Popescu, Alexandru Petrescu* (în prezent în Luxemburg), *Daniel Pandelea, Ştefan Puşcaşu* şi *Laurenţiu Badea* (în prezent în SUA) au făcut minuni. Actualii administratori *Bogdan Brătucu, Victor Stănescu,* studenţi la Facultatea de Calculatoare, fac eforturi lăudabile pentru a întreţine o configuraţie hardware–software extrem de complexă.

Un impact deosebit nu numai în LMN ci în întreaga comunitate Unix din România au avut-o: programul "*Free Unix for Romania*" (iniţiat de *Marius Hâncu* din Canada şi continuat de *Teodor Lungu* de la NASA-JPL-Passadena) prin care a fost difuzat sistemul Linux, precum şi organizarea de către *GURU* – "*Grupul Utilizatorilor Români de Unix*" a conferinţei anuale *ROSE – Romanian Open System Event* (în care un rol hotărâtor l-au avut ing. *Alexandru Rotaru* şi prof. *Irina Athanasiu*), ocazie cu care am fost vizitaţi de cele mai mari personalităţi din domeniu, cum sunt *Richard Stalman* ("Mr GNU") şi *Linus Torvald* ("Mr Linux").

Interesul LMN pentru educaţia asistată de calculator a inginerilor datează din 1992, când Catedra de Electrotehnică a fost dotată de Ministerul Învăţământului (condus în acea vreme de prof. *George Ştefan*) cu o reţea Novell cuprinzând un server PS/2 (386) şi 12 calculatoarea PS/2 (286) cu ecran monocrom, pentru a fi folosită pentru studenţii anului al II-lea la laboratorul asociat cursului de Metode Numerice. Într-un timp record (continuând preocupările începute în 1991), asistenţii acestui curs: *Irina Munteanu, Bogdan Ionescu* (în prezent angajat la cea mai mare firmă pentru software de modelare electromagnetică, ANSOFT din SUA), *Radu Popa* (în prezent doctorand la Universitatea din Tokio), *Mihai Lăzărescu, Mihai Popescu,* dar la care s-au asociat ulterior *Gabriela Ciuprina, Mihai Platon, Mihai Rebican, Gabriel Preda* şi *Cristina Bordeanu,* au dezvoltat un pachet de programe dedicat instruirii asistate în domeniul Metodelor Numerice în Ingineria Electrică. Acest pachet s-a bucurat nu numai de impact local, ci şi de un deosebit succes internaţional. În acest fel ne-am câştigat noi suporteri: *Jean Michel* (de la renumita Şcoală de Poduri şi Şosele din Paris), *Eddy Forte* şi *Guy de Lafontaine* (de la Şcoala Politehnică Federală din Lausanne).

Apariţia tehnicilor multimedia şi încercarea de a fi folosite în educaţie prin producerea de materialele didactice de către ingineri şi studenţi în inginerie nu a dat rezultate strălucite

datorită aspectului neprofesional al acestora. Acesta este motivul pentru care LMN s-a implicat din 1995 în proiectul *TEMPUS S-JEP 9122,* dedicat Comunicării Audiovizuale bazate pe Tehnologia Informaţiei (cu un buget de peste 500.000 ECU) şi în care, pe lângă U.P.B., au mai participat: Academia de Teatru şi Film (prin Catedra de Multimedia reprezentată de coordonatorul proiectului – *Horea Murgu,* actualul Rector al Academiei – *Florin Mihăilescu,* iar în faza iniţială *Victor Rebengiuc* şi *Stere Gulea* – ulterior preşedintele interimar al Televiziunii Române), Academia de Arte Vizuale din Cluj (reprezentată de şeful secţiei de Foto-Video, *Karoly Feleky* şi de Rectorul Academiei, *Cornel Nica*).

Împreună cu personalităţile din lumea artistică menţionate şi cu partenerii noştri de peste hotare: *Roy Ascott* (Newport, UK), *Yossi Bal* (Londra, UK), *Hans Froling* şi *Peter van der Weyden* (Utrecht, NL), *Michèle Ganem* (Grenoble, F), *Yves Papin* şi *Guy Bozard* (Nantes, F), Geneviève Toussaint (Bruxelles, B), am făcut un pas înainte în înţelegerea tainelor comunicării profesionale. CD-ROM-ul "Bucureşti 2000", realizat în LMN şi primul curs de Comunicare Profesională adresat inginerilor din România, reprezintă dovezi incontestabile în acest sens. Interesul pentru comunicarea orală sau scrisă dar mai ales pentru comunicarea electronică (Internet, Web) reflectate în manualul de faţă reprezintă efecte ale acestui proiect.

De altfel, pe lângă apariţia acestui manual, proiectul *TEMPUS S-JEP 9122* a mai contribuit şi la dotarea laboratorului de comunicare profesională (EB-212). Tot în cadrul acestui proiect, partenerii noştri olandezi au donat Universităţii "Politehnica" din Bucureşti reţeaua de staţii grafice VAX din EB-211. Ne face plăcere să le mulţumim pentru generozitatea lor şi pentru energia depusă în organizarea selecţiei şi a transportului echipamentelor. Cu această ocazie au fost donate calculatoare şi Centrului pentru Promovarea Comunicării coordonat de studenţii Facultăţii de Electrotehnică din UPB. Caracterul absolut novator pentru România al acestui proiect a generat interes şi sprijin atât din partea unor personalităţi precum *Andrei Marga* (Rectorul Universităţii Babeş Bolyai, în prezent Ministrul Învăţământului), dar şi unor organizaţii ca *Fundaţia Soros pentru o Societate Deschisă*, care a cofinanţat proiectul cu 120.000 USD.

Nu trebuie să uităm că toate aceste activităţi s-au desfăşurat în Catedra de Electrotehnică (ce a fost condusă din 1990 de profesorii *Andrei Ţugulea, Aurelian Panaitescu* şi *Florea I. Hănţilă*), a Facultăţii de Electrotehnică (ce a fost condusă de Decanii *Aurelian Crăciunescu, Constantin Răduţi* şi *Mihai Popescu*), din U.P.B. (ce a fost condusă de Rectorii *V.N. Constantinescu* şi *Gh. Zgură*). Tuturor le mulţumim pe măsura sprijinului acordat.

Nu mai puţin le mulţumim studenţilor care s-au perindat prin LMN şi fără de care eforturile noastre nu ar fi avut nici un sens. Mulţumiri merită şi personalul tehnic (*Cornel Niţulescu, Ion Ioniţă, Mihai Sandu, Mary Klein, Nicolae Boboc, Flavian Colonel, Gabriel Ioan*), dar cu prisosinţă doamnei *Gabriela Iosif,* persoană a cărui devotament şi dăruire pentru LMN este peste orice îndoială.

Suntem convinşi că lista persoanelor care au contribuit într-un fel sau altul la dezvoltarea laboratorului şi au făcut posibilă apariţia acestui manual nu este completă.

Am lăsat la urmă familiile noastre care ne-au iertat pentru weekend-urile, serile şi chiar nopţile pe care li le-am răpit.

Capitolul 1

Utilizarea sistemului de operare Unix

Sistemul de operare este cea mai importantă componentă sofware a unui sistem de calcul. El dă viaţă echipamentului (componentei hardware) asigurând următoarele **funcţii principale**:

- gestiunea spaţiului de **memorie** (alocări şi eliberări de zone în memoria internă RAM);
- gestiunea **proceselor** (lucru extrem de important mai ales în sistemele care pot executa "simultan" mai multe programe);
- gestiunea spaţiului de **disc** (prin intermediul sistemului de fişiere);
- controlul operaţiilor de **intrare–ieşire** (operaţii elementare ca de exemplu accesul direct în memorie –DMA– sau operaţii mai complicate realizate prin intermediul unor componente software numite "drivere", specifice fiecărui periferic: tastatură; mouse; ecran; imprimantă; unităţi de bandă; CD-ROM sau alte dispozitive de memorare externă; interfeţe şi plăci de comunicaţie, etc.).

Operaţiile fundamentale ale sistemului de operare sunt incluse în "*nucleul*" lui şi nu sunt de obicei accesibile direct de utilizator, din motive de securitate. **Interacţiunea** dintre nucleul sistemului de operare şi utilizator se realizează printr-o altă componentă a sistemului, numită "*shell*", un interpretor de comenzi care preia, interpretează şi execută comenzile utilizatorului, introduse de obicei de la tastatură. Sistemele moderne mai au o componentă suprapusă peste shell, numită *interfaţă grafică cu utilizatorul* (*GUI – Graphic User Interface*), care permite interacţiunea calculator–utilizator prin intermediul unui **sistem de ferestre** pe ecran, însoţite de butoane şi meniuri selectabile prin mouse sau de la tastatură.

Istoria evoluţiei calculatoarelor este strâns legată, pe de o parte, de evoluţia tehnologiilor de realizare a componentei hardware (mai ales a componentei principale care este procesorul), iar pe de altă parte, de evoluţia componentei software, în care sistemul de operare joacă un rol central. Datorită importanţei lui, practic fiecare firmă producătoare

de calculatoare şi-a dezvoltat şi întreţinut propriul sistem de operare. Odată cu apariţia şi răspândirea ca bun de larg consum a calculatorului personal *PC* (*IBM PC* şi compatibile) situaţia s-a schimbat. Marea majoritate a PC-urilor folosesc în prezent sistemul de operare *MS–Windows*, dezvoltat de firma *Microsoft* pe baza unui sistem anterior numit *MS–DOS* (Microsoft Disk Operating System).

Pe al doilea loc, ca număr de utilizatori, se află sistemele de operare de tip Unix [65] [73]. Acestea au fost dezvoltate iniţial pentru minicalculatoare (*DEC PDP*), dar au fost ulterior intens folosite de staţiile de lucru (sistem de calcul cu procesoare puternice *RISC*) produse de firmele *Sun, Silicon Graphics, Digital Hewlett Packard* sau *IBM.* Începând cu apariţia procesoarelor *Intel 386, 486* şi *Pentium*, sistemele de tip Unix au migrat şi în lumea PC-urilor. Trebuie remarcat faptul că sistemele Unix au fost permanent mai avansate conceptual şi ca soliditate, decât sistemele DOS- Windows, fiind dedicate unor utilizări profesionale.

UNIX a devenit operaţional ca sistem de operare pentru *DEC PDP 11* în februarie 1971. El a fost dezvoltat iniţial de o echipă mică de programatori (D. Ritchie şi K. Thomson) de la *Bell Laboratories* (*AT&T*), prin efort personal (de circa 2 ani–om) şi nu pe baza unor specificaţii predefinite, sarcini de servici, cum s-a întâmplat cu toate sistemele anterioare. Bazaţi pe un sistem numit *Multics* (dezvoltat fără mare succes în anii '60 de *MIT, Bell* şi *General Electric*) şi nemulţumiţi de ospitalitatea sistemelor existente, ai au decis să-şi scrie propriul sistem de operare, mic ca dimensiuni, dar puternic, şi care să implementeze cele mai fertile idei cunoscute în acel moment. Sursele sistemului de operare au fost distribuite gratis în comunitatea academică americană şi au fost folosite în aplicaţii referitoare la educaţia în ştiinţa calculatoarelor (sisteme de operare, limbaje, reţele de calculatoare), dar şi pentru editare texte şi formatare de documente.

În perioada 1974 - 1979 UNIX a fost folosit intensiv şi practic rescris din nou de studenţii universităţii Berkley. Versiunea finală a fost *BSD 4.4* (*Berkley Software Distribution*). În paralel, cei de la Bell au ajuns la versiunea Unix *System V.* Prin reunirea practic a celor două versiuni, în 1985 s-a obţinut produsul numit *UNIX SVR4* (System V Release 4). În acel moment, sistemul avea câteva mii de instalări şi deţinea un compilator de C, asamblor, editor de legături, depanator simbolic, editor de texte, sistem de formatare a documentelor, sute de instrumente diverse şi compilatoare pentru mai multe limbaje incluzând *FORTRAN 77, Basic* şi *Pascal.*

Experienţa absolvenţilor Berkley a migrat odată cu ei la marile firme de calculatoare din Valea Siliciului, astfel încât au apărut pe piaţă o multitudine de sisteme de tip Unix: *HP-UX* (Hewlett Packard) [35], *Ultrix* (Digital) [34], *IRIX* (Silicon Graphics) [32], *Xenix* (Microsoft), *SCO* (filiala Microsoft: Santa Cruz Organisation) [47] şi la compania Sun: *Sun OS* (de tip BSD) şi *Solaris* (de tip SVR9) [38]. După ce a aparţinut o vreme unui laborator nou creat în AT&T: *USL* (UNIX System Lab.), marca **UNIX** a fost vândută companiei *Novel* (care distribuie *UNIXWare*) şi în final a ajuns să aparţină unei organizaţii de standardizare numită *X/Open.*

Încercările de standardizare au început cu înfiinţarea *Open Software Foundation* sprijinită de *Digital* (distribuitorul unei versiuni de sistem de operare numit chiar *OSF/1*) şi *IBM.* În prezent, efortul de standardizare este realizat de IEEE care a elaborat primele două

capitole din standardul *IEEE 1003*, numit şi *POSIX* (*Portable Operating System Interface*) şi care în 1003.1 descrie funcţiile necesare programării în C, iar în 1003.2 descrie setul de comenzi utilizator [9].

Deoarece era convins că produsele software trebuie să fie gratuite, un tânăr şi extrem de talentat programator, absolvent MIT, numit Richard Stallman, a înfiinţat în 1990 o fundaţie numită *Free Software Foundation*. Principalul proiect al acestei fundaţii este numit *GNU* (*GNU = GNU is Not Unix*) [6] şi urmăreşte realizarea unui nou sistem de operare care să satisfacă prevederile POSIX şi a unui set complet de instrumente software de foarte bună calitate. Produsele GNU nu sunt plasate în domeniul public ("*public software*") şi nu sunt nici promoţionale ("*shareware*"), ci sunt supuse unui contract de copiere numit *GPL* ("*General Public Licence*"), care în esenţă stipulează că oricine are dreptul să vândă pachetul de software la orice preţ (indiferent dacă l-a obţinut gratis!), cu condiţia să livreze şi codul sursă. În ciuda reticenţei cu care au fost privite FSF, GNU şi GPL, se constată că proiectul reuşeşte (aducând inclusiv avantaje financiare!).

Un alt eveniment important în lumea Unix are loc în 1991 când tânărul student finlandez Linus Torvald începe, practic de la zero, (şi influenţat de sistemul de şcoală *Minix* al lui Tanenbaum), dezvoltarea unui nou sistem de operare de tip Unix pentru PC 386, sistem pe care l-a numit *Linux* [2] [10] [11] [12] [16] [21] [40]. Istoria este interesantă deoarece sistemul Linux este dezvoltat şi întreţinut de o armată uriaşă de programatori din întreaga lume, pe bază de voluntariat, folosind Internet-ul. În prezent, sistemul GNU Linux (cu clauza GPL) este cel mai utilizat sistem de tip Unix din lume, având milioane de instalări.

Caracteristicile sistemelor Unix, care explică succesul lor, sunt:

- **portabilitatea**, fiind accesibil în sursă C poate şi este rulat pe o mare varietate de calculatoare;
- **puterea**, deoarece este acompaniat de o multitudine de instrumente şi programe de aplicaţii, care permit creşterea productivităţii în cele mai diverse domenii, de la birotică la cercetare ştiinţifică;
- **multitasking**, permite executarea "simultană" (de fapt în timp partajat) a mai multor operaţii, ca de exemplu redactarea unui text şi în background formatarea unui document sau/şi rezolvarea unei ecuaţii;
- **multiuser**, permite accesul simultan la resurse valoroase (imprimante, calculatoare puternice sau servere de fişiere) a mai multor utilizatori;
- **conectivitate**, sistemul având suport nativ pentru conectarea calculatoarelor în reţea (networking).

În afara calităţilor menţionate, utilizarea sistemelor Unix întâmpină şi unele dificultăţi legate de diversitatea variantelor şi versiunilor (reflectată de altfel şi în LMN, în care sunt instalate pe lângă sistemul Linux şi variante comerciale ca HP-UX, SunOS, Ultrix

şi Irix) şi de faptul că sistemul de instrumente Unix nu este foarte coerent (apar dese suprapuneri ale funcţiilor, iar interfaţa lor cu utilizatorul nu este comună).

Începând încă din anii '70, firma *Rank Xerox* a finanţat un proiect de cercetare care să conducă la perfecţionarea relaţiei om-calculator. Ca urmare a acestui proiect (Xerox PARC) s-a inventat mouse-ul şi conceptele specifice interfeţelor grafice cu utilizatorul – GUI. Bazat pe aceste concepte s-a realizat la universitatea Berkley un sistem de ferestre (numit W), care a fost apoi preluat (în proiectul Athena) de MIT şi care a condus în final la realizarea celei mai populare interfeţe grafice pentru sistemele Unix, numită *XWindow* [49] [72]. Ultima versiune majoră a acestui sistem, numită pe scurt *X11*, a fost realizată în 1987 şi este distribuită gratis, începând din 1988, de *MIT X Consortium*.

Cele mai importante caracteristici ale sistemului XWindow sunt:

- **portabilitatea**, fiind disponibil în sursă C;
- **transparenţa**, fiind dezvoltat în maniera client/server, el poate fi exploatat în reţeaua de calculatoare urmând ca să fie posibil ca un program să ruleze pe o maşină, folosind fişierele de pe altă maşină, iar rezultatele să le afişeze într-o fereastră de pe o a treia maşină;
- **flexibilitatea**, pot fi deschise oricâte ferestre, stilul ferestrelor nu este predefinit şi suportă controlul unui număr nedefinit de managere de ferestre ca: *Athena, TWM, OpenLook* sau *Motif* (care în prezent este cel mai răspândit, dar nu este încă disponibil gratis).

Trebuie menţionat că managerul de ferestre impune stilul ("look and feel") ferestrelor generate.

Scopul principal al capitolului constă în familiarizarea cu comenzile sistemului de operare Unix şi a interfeţelor XWindow privitoare la sesiunea de lucru (intrare, ieşire, securitate, configuraţie), sistemul de fişiere (creere, redenumire, ştergere, copiere, căutare, arhivare, protecţie), vizualizare conţinut fişiere, filtre şi redirectări, imprimare şi documentare.

1.1 Intrarea în sistem

Pentru a putea lucra în sistemul Unix trebuie să aveţi un "cont" pe acel sistem. Fiecărui utilizator cu un astfel de cont i se acordă de către administratorul de sistem:

- un nume de cont (*username*);
- o parolă (*password*).

Iniţial, pe ecranele calculatoarelor este afişată **fereastra de acces în reţea**, prin care se poate iniţia o sesiune de lucru. Fereastra are aspectul:

Welcome to Kaiser

Username:

Password:

În această fereastră se introduc numele de cont şi parola, fiecare urmate de <↵> (Enter). Din motive de securitate, parola nu este afişată pe ecran.

Dacă numele şi parola sunt recunoscute de sistemul de operare, după scurt timp pe ecran se deschide o fereastră XWindow.

1.2 Lucrul cu ferestre XWindow

La începutul unei sesiuni de lucru, sistemul afişează o fereastră de lucru (XTerm) şi un grup (meniu) de butoane (să-l numim în continuare Toolbox) în colţul din dreapta jos a ecranului. Meniul Toolbox este descris în anexa D.3.1.

Mutarea cursorului mouse–ului în fereastra de lucru are ca efect vizibil schimbarea culorii barei superioare a ferestrei. Se marchează astfel faptul că fereastra a devenit **fereastra activă** de lucru. În acest moment se pot da comenzi la promptul sistemului, cum ar fi de exemplu `uname` dacă vreţi să aflaţi ce variantă de Unix folosiţi, sau `hostname` pentru a afla numele maşinii pe care lucraţi (figura 1.1).

Butoanele mouse–ului au funcţii diferite, după cum cursorul se află plasat într-o fereastră sau în afara oricărei ferestre.

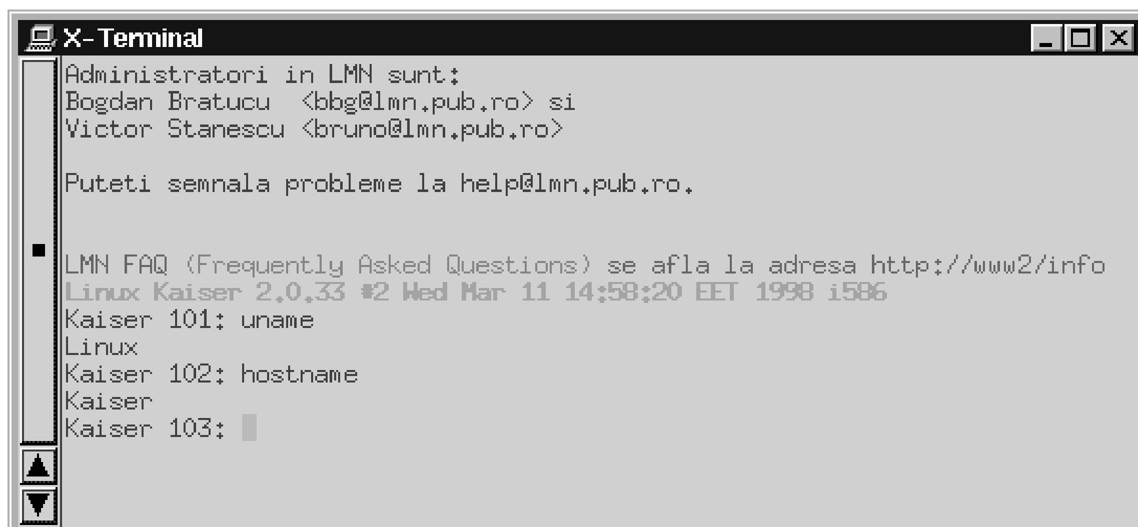


Fig. 1.1: Fereastră XWindow (XTerminal)

- **Când cursorul mouse–ului este plasat într-o fereastră**
  - Butonul din stânga: permite *selectarea* unui text, astfel: se plasează cursorul la începutul textului, se apasă butonul din stânga şi se deplasează mouse–ul ţinând butonul apăsat. Textul astfel selectat îşi schimbă culoarea.
  - Butonul din mijloc: permite *copierea* la promptul sistemului a textului selectat, ca şi cum el ar fi introdus de utilizator de la tastatură literă cu literă.
  - Butonul din dreapta: se comportă similar cu butonul din stânga, dar permite *selectarea* unei porţiuni mai mari de text: se deplasează cursorul la începutul textului de selectat, se apasă scurt butonul din stânga (*click*!) fără a deplasa mouse–ul, după eliberarea butonului se deplasează mouse–ul la sfârşitul textului de selectat, se apasă acum butonul din dreapta.

- **Când cursorul mouse–ului este plasat pe bordura de sus a unei ferestre**
  - Butonul din stânga: permite *mutarea ferestrei*;
  - Butonul din dreapta: permite *ridicarea ferestrei* la suprafaţă (deasupra altor ferestre de pe ecran) sau *ascunderea* ei (sub celelalte ferestre).

- **Când cursorul mouse–ului este plasat în afara oricărei ferestre** (pe fundalul ecranului)
  - Butonul din stânga: afişează un **meniu** din care se pot *lansa în execuţie* diferite programe sau utilitare. Dintre acestea, deosebit de utile sunt: prima linie a meniului, care permite *deschiderea unei noi ferestre de lucru*; ultima linie a meniului, care permite *ieşirea din sistem* (nu încercaţi această opţiune chiar în acest moment!).

- Butonul din mijloc: permite efectuarea de operaţii cu ferestrele de lucru: schimbarea *dimensiunilor* (resize), *mutarea* (move) şi *închiderea* (kill) ferestrelor.
- Butonul din dreapta: afişează o listă a ferestrelor deschise, şi permite selectarea unei ferestre şi implicit deplasarea rapidă a cursorului în *fereastra activată.*

- **Când cursorul mouse–ului este plasat pe bordura laterală a unei ferestre**

  - Cu butonul din stânga se poate agăţa un colţ al ferestrei şi să se *mărească* sau *micşoreze* dimensiunile ferestrei.
  - **Observaţie**: Schimbarea dimensiunii unei ferestre la dimensiunea maximă (cât tot ecranul) se poate face şi "apăsând" (cu ajutorul butonului din stânga al mouse-ului) al doilea buton (pătrăţelul mare) aflat în partea din dreapta sus a bordurii ferestrei.

Dacă la un moment dat sunt deschise mai multe ferestre, în fiecare dintre ele se pot da comenzi independente, ca şi cum utilizatorul ar avea la dispoziţie mai multe terminale (ecrane). Gestionarea unui număr mare de ferestre (care la un moment dat pot să se suprapună, unele dintre ele nefiind vizibile) este facilitată de:

- bara orizontală plasată în partea de jos a ecranului; ea conţine câte un buton pentru fiecare fereastră deschisă; apăsarea unui astfel de buton are ca efect mutarea cursorului în fereastra corespunzătoare şi selectarea acesteia ca fereastră activă;

- cele trei căsuţe intitulate Desktop din Toolbox (grupul de butoane plasate în colţul dreapta jos al ecranului, fig. 1.2); să notăm căsuţele cu numere de ordine: Desktop0, Desktop1, Desktop2. Ele pot fi privite ca porţiuni din "ecranul logic" pe care se pot plasa ferestre. Numai una din cele trei părţi ale acestui "ecran logic" este afişată, la un moment dat, pe ecranul fizic (la început, Desktop0).

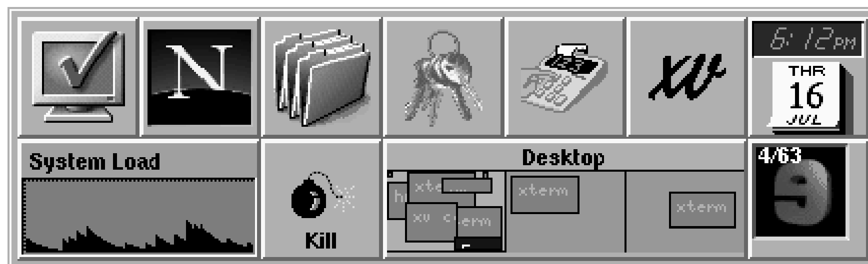


Fig. 1.2: Toolbox

1.3 Ieşirea din sistem

- Se închid pe rând toate ferestrele de lucru deschise, de exemplu folosind comanda `exit`; acest pas nu este obligatoriu, dar evită eventuala pierdere a unor date nesalvate;
- Se apasă butonul din stânga al mouse–ului (cursorul acestuia fiind plasat pe fundal, nu într-o fereastră) şi se selectează ultima linie din meniu (`Exit Fvwm95`);
- La noul meniu apărut se răspunde (prin selectare cu mouse–ul) `Yes, really quit`.

Încheierea sesiunii de lucru este obligatorie când părăsiţi laboratorul, în caz contrar riscaţi ca la următoarea şedinţă sa'nu mai fiţi recunoscut ca utilizator al sistemului!

1.4 Sesiunea de lucru şi configurarea ei

- `login`

  Permite accesul unui utilizator în sistem de la o consolă a sistemului. Se solicită introducerea numelui de utilizator şi a parolei.

- `rlogin` *nume_maşină* (remote login), `ssh` *nume_maşină* (secure shell)

  Permite accesul unui utilizator la un sistem aflat "la distanţă" (în altă reţea decât reţeaua locală la care utilizatorul este conectat).

  Parametrul *nume_maşină* reprezintă numele calculatorului aflat la distanţă. Programul solicită introducerea parolei. În mod implicit, se consideră că numele de utilizator de pe maşina de la distanţă este acelaşi cu numele de utilizator din reţeaua locală.

  Dacă numele de utilizator pe maşina de la distanţă este altul decât cel curent, trebuie adăugat încă un parametru, sub forma:

  `ssh` *nume_maşină* `-l` *nume_utilizator*

  Se recomandă folosirea comenzii `ssh` deoarece transmite parola criptată şi nu în clar, fiind astfel mai sigură decât `rlogin`

- `exit`

  Opreşte execuţia interpretorului de comenzi Unix şi încheie sesiunea de lucru.

  În cazul lucrului într-un sistem bazat pe ferestre, are ca efect închiderea ferestrei (într-un astfel de sistem se lansează câte un interpretor de comenzi în fiecare fereastră deschisă). În cazul lucrului cu o singură consolă de ieşire, are ca efect ieşirea din sistem.

- `passwd` – permite schimbarea parolei.

  Se solicită introducerea vechii parole şi apoi a celei noi (de două ori, pentru verificarea introducerii corecte a noii parole).

  Parola trebuie să fie suficient de lungă (numărul minim de caractere variind de la sistem la sistem), să conţină atât litere cât şi caractere speciale (numere, semne de punctuaţie).

  Recomandări: Schimbaţi parola periodic. Alegeţi-vă o parolă pe care să o ştiţi numai Dvs., să fie uşor de memorat, deoarce dacă aţi uitat-o nu mai puteţi intra în sistem. Nu ignoraţi mesajul afişat la intrarea în sistem, care vă poate avertiza în legătură cu data de expirare a parolei. Nu comunicaţi nimănui parola şi nu o notaţi.

- `lock` sau `xlock` – blochează terminalul până la introducerea parolei corecte a utilizatorului.

  Vă recomandăm să folosiţi această comandă de câte ori părăsiţi locul de lucru din faţa calculatorului, pentru a evita folosirea contului Dvs. de către persoane neautorizate.

  În reţeaua din sala EB212, această comandă este apelată în mod automat dacă nu daţi nici o comandă de la tastatură sau prin mouse, timp de câteva minute.

  Ţineţi cont de faptul că în lipsa Dvs. alţi utilizatori pot avea nevoie să lucreze în sistem, deci, dacă intenţionaţi să lipsiţi mai mult de cinci minute, închideţi sesiunea de lucru (în caz contrar riscaţi penalizări din partea administratorului de sistem, cum ar fi suspendarea contului pe o perioadă).

- `hostname` – afişează numele maşinii pe care lucraţi.

- `uname` – afişează numele sistemului de operare folosit.

Pentru utilizări avansate:

- `printenv` – afişează lista variabilelor de configurare a sesiunii de lucru.

  Valorile acestor variabile se pot schimba cu:

- `setenv` *varibila valoare* (în shell-ul csh)

  `set` *variabila=valoare*
  `export $variabila` (în shell-ul sh)

  `declare` *variabila=valoare* (în shell-ul bash)

    Exemplu: `declare -x TMP="/home/irina/tmp"`

  Aceste comenzi sunt specifice **shell**-ului şi nu sistemului de operare **Unix**!

  Comenzile de configurare pot fi scrise în fişiere de configurare specifice shell-ului. Numele acestor fişiere începe cu `.` urmat de numele shell-ului, urmat de literele **rc**.

Exemplu: `.bashrc` – pentru shell-ul bash

`.cshrc` – pentru shell-ul csh

Printre variabilele predefinite se numără: **HOME, SHELL, TERM, LOGNAME, PATH, MAIL, TZ**.

Odată definite, valoarea unei variabile se obţine punând semnul $ înaintea numelui acesteia.

Exemplu: `echo $TMP` – afişează valorea curentă a variabilei de configurare TMP.

1.5 Interpretorul de comenzi

În sistemele Unix interpretoarele de comenzi ale utilizatorului (shell) sunt de fapt tratate de sistemul de operare ca şi orice alt program executabil. În consecinţă, există o mare diversitate de astfel de interpretoare care se pot împărţi în două categorii:

- cele specifice Unix System V (la care prompterul este $):
  - Bourne Shell: `sh`
  - Korn Shell: `ksh`
  - GNU Bourne Again Shell: `bash`
- cele specifice sistemelor de tip BSD (la care prompterul este %), cel mai cunoscut fiind Cshell: `csh`.

După cum aţi văzut în paragraful anterior, `ssh` (Secure SHell) are un rol aparte, permiţând conectarea de la distanţă.

În mod normal, comenzile către shell sunt date de la tastatură, dar ele pot fi preluate şi dintr-un fişier. Un **script** este un fişier care conţine comenzi ale shell-ului, utilizat pentru a executa succesiuni lungi de comenzi Unix sau pentru a scrie scurte "programe" care sunt apoi apelate prin numele de fişier şi sunt tratate la fel cu orice comandă Unix. În acest fel se obţine, practic, o extindere de către utilizator a comenzilor recunoscute de sistem.

1.6 Comenzi pentru lucrul cu fişiere şi directoare

1.6.1 Structura de directoare şi fişiere

Ca orice sistem modern de operare, Unix operează cu două concepte legate de stocarea ierarhizată a informaţiei:

- **fişiere**: entităţi care conţin informaţii (text, instrucţiuni de programare);
- **directoare**: entităţi care conţin fişiere şi care permit gruparea şi ordonarea acestora.

Folosind similitudinea cu o bibliotecă, fişierele se aseamănă cu cărţile, în timp ce directoarele se aseamănă cu rafturile în care acestea sunt aşezate. La rândul lor, rafturile pot fi grupate în dulapuri de cărţi, care în Unix sunt tot directoare. Fiecare "dulap de cărţi", raft sau carte au un nume. Pentru a preciza unde se află o carte, trebuie precizate toate informaţiile care să permită găsirea cărţii: dulapul, raftul, titlul. Tot astfel în Unix trebuie precizată toată *calea* pe care se poate ajunge la un director sau fişier. Calea reprezintă o succesiune de nume de directoare (eventual ultimul nume din succesiune putând fi un nume de fişier), separate prin caracterul `/` (în limba engleză *slash*).

Numele fişierelor şi directoarelor pot conţine litere, cifre, caractere speciale: `_` (underscore sau caracter de subliniere), `-` (minus) şi `.` (punctul), etc. Spre deosebire de DOS, numele de fişiere pot conţine oricâte puncte.

Exemplu:

`program1, tema.pas.bak, tema-1, tema_2`

sunt nume valide de fişiere sau de directoare.

Caracterul * înlocuieşte oricâte caractere din numele unui fişier. Caracterul ? înlocuieşte un singur caracter.

Exemple:

`te*a`

se poate referi la oricare din următoarele nume de fişiere: `tema, tematica, teza, tt.c.bak.a`

`te?a`

se poate referi la nume de fişiere precum `tema, teza, te.a`, dar nu şi la `tematica, tt.c.bak.a`

Un grup de caractere de tipul `[a-z]` selectează un singur caracter, cuprins (în lista codurilor ASCII) între `a` şi `z`.

Exemplu:

`[g-i]*`

selectează toate fişierele al căror nume începe cu litera `g` sau `h` sau `i`.

Directoare "speciale"

- **directorul curent** – este simbolizat printr-un singur singur punct.

- **directorul precedent** în arborele de directoare – este simbolizat prin .. (de două ori punct).
- **directorul rădăcină** – cel care conţine toate celelalte directoare şi fişiere ale sistemului, este simbolizat prin / (*slash* în limba engleză).
- **directorul home** – cel în care este plasat utilizatorul la intrarea în sistem, este simbolizat prin `$HOME` sau prin ~. În principiu, un utilizator are drepturi depline (de citire şi de scriere) doar în directorul home şi în toate directoarele care se află în acesta.

În figura 1.3 se prezintă o parte din structura tipică a directoarelor unui sistem Unix.

Pe calculatoarele PC din sala EB212 este instalat atât Linux cât şi Windows95 (intrarea în Windows făcându-se din Linux cu comanda `dos`). Aceasta este explicaţia prezenţei directoarelor nestandard `/dos/c/` şi `/dos/d/`, care vă permit accesul din Linux la fişierele create sub sistemele de operare DOS şi Windows.

În sistemul prezentat în figura 1.3, dacă numele Dvs. de utilizator este `ioan`, directorul home este `/home/elth/an6/ioan`.

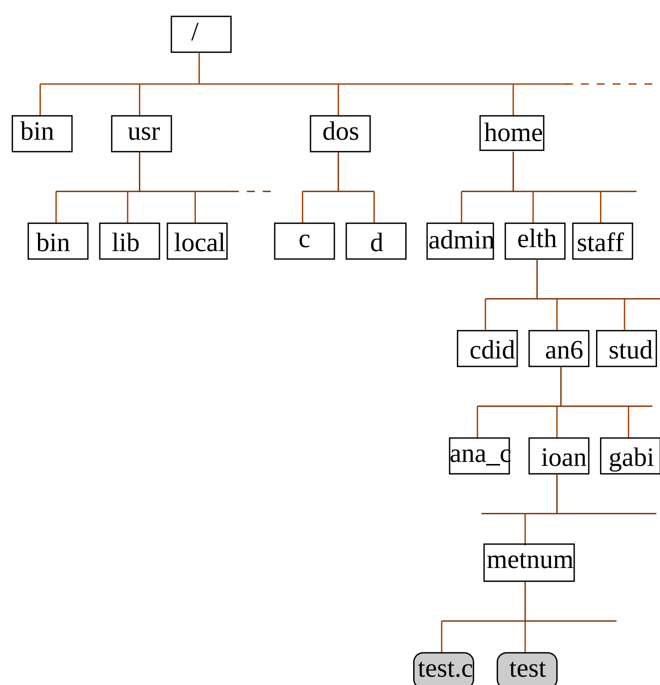


Fig. 1.3: Structura directoarelor unui sistem Unix

Vă recomandăm ca pentru fiecare din proiectele Dvs. să folosiţi un director dedicat.

Pentru a avea acces la un fişier trebuie să specificaţi calea, ca în exemplele:

- absolut:

  `/home/elth/an6/ioan/metnum/test.c`

- relativ (presupunând că vă aflaţi în directorul /home/elth/an6/ioan/):

  metnum/test.c

1.6.2 Comenzi pentru lucrul cu fişiere

- ls – afişează conţinutul (lista directoarelor şi fişierelor) directorului curent.

  Comanda ls poate fi apelată cu parametri, de exemplu:

  ls -l

  pentru afişarea conţinutului directorului curent, cu informaţii detaliate privitoare la drepturile de citire (r), scriere (w) sau execuţie (x) asupra fişierului, data şi ora ultimei modificări, dimensiunea fişierului, etc.

  ls -a

  pentru listarea tuturor fişierelor din directorul curent, inclusiv cele al căror nume începe cu punct (prin convenţie, numele fişierelor "ascunse" ale sistemului de operare Unix încep cu punct).

- cat *nume_fişier* – listează conţinutul unui fişier.

- more *nume_fişier* – listează conţinutul unui fişier, cu oprirea afişării după fiecare pagină.

  După lansarea more, sunt active următoarele comenzi:

  - *spaţiu* – avansare cu o pagină spre sfârşitul fişierului;
  - b – avansare cu o pagină spre începutul fişierului;
  - < ↵ > – avansare cu un rând înainte;
  - q – ieşire din more.

- cp *nume_fişier_existent* *nume_nou* – copiază fişierul *nume_fişier_existent* sub numele *nume_nou*.

- mv *nume_fişier_existent* *nume_nou* – redenumeşte fişierul *nume_fişier_existent*.

  mv *nume_fişier_existent* *nume_director* – mută fişierul în directorul *nume_director*.

- rm *nume_fişier_existent* – şterge fişierul.

  Comanda rm poate fi apelată cu parametri. De exemplu, un parametru deosebit de util este -i:

  rm -i nume_fişier_existent – şterge fişierul cu solicitarea unei confirmări.

1.6.3 Comenzi pentru lucrul cu directoare

- `cd` *nume_director* – schimbă directorul curent, în care *nume_director* este o cale absolută sau relativă către un director.

  Exemplu: Efectul următoarelor trei comenzi:

`cd /`	*schimbă directorul curent în directorul rădăcină*
`cd stiinta`	*schimbă directorul curent în* `/stiinta` *(deplasare relativă)*
`cd matematica`	

  este echivalent cu al comenzii:

  ```
  cd /stiinta/matematica
  ```

  Comanda `cd` fără nici un nume de director are ca efect mutarea în directorul *home*.

- `pwd` – afişează numele directorului curent (calea absolută).

- `mkdir` *nume_director* – crează un nou director. Dacă numele directorului nu este precedat de o cale, noul director este creat în cel curent.

  Exemplu:

  `mkdir modelare` – crează directorul `modelare` în directorul curent

  `mkdir /modelare` – crează directorul `modelare` în directorul rădăcină (de obicei, această comandă va genera un mesaj de eroare, deoarece nu aveţi drepturi de scriere în directorul rădăcină).

- `rmdir` *nume_director* – şterge directorul.

  Dacă directorul conţine alte fişiere sau directoare, el nu poate fi şters cu comanda anterioară.

  `rm -rf` *nume_director* – şterge recursiv (`-r`) şi forţat (`-f`) toate fişierele şi directoarele aflate în directorul *nume_director*.

  Atenţie! Comenzile `rm` şi `rmdir` au efect definitiv! Un fişier sau director, odată şters, nu mai poate fi recuperat!

1.6.4 Schimbarea drepturilor asupra fişierelor/directoarelor

- `chmod` *abc nume*

  Schimbă drepturile asupra fişierului sau directorului *nume*, conform cu parametrii *abc*.

  Grupul celor trei litere semnifică: prima literă, al cui drept este modificat; a doua, în ce sens este modificat (acordat sau retras); a treia, care drept este modificat (de citire, de scriere sau de execuţie).

  a – poate lua valorile: `u` –utilizatorul; `g` –grupul din care face parte utilizatorul; `o` –toţi ceilalţi utilizatori.

b – poate lua valorile: `+` –acordarea dreptului; `-` –retragerea dreptului.

c – poate lua valorile: `r` –drept de citire; `w` –drept de scriere (modificare); `x` –drept de execuţie.

Exemplu: `chmod u+x test1` – acordă drept de execuţie proprietarului fişierului `test1` (`test1` devine un fişier executabil ca orice comandă Unix).

- `umask` *cod*

  Schimbă drepturile implicite asupra fişierelor conform codului *cod*.

1.6.5 Căutări

- `locate` *nume*

  găseşte toate fişierele şi directoarele al căror nume include *nume*. Căutarea se face în toate directoarele sistemului în care utilizatorul are drept de citire.

- `find` *cale opţiuni*

  găseşte toate fişierele care satisfac condiţiile din *opţiuni*, care se află în directorul *cale* şi în toate subdirectoarele acestuia.

  Exemplu: `find /usr -name "test*" -print`

  caută recursiv, în directorul `/usr` şi în subdirectoarele acestuia, toate fişierele/directoarele al cărui nume începe cu `test` şi afişează rezultatul pe ecran. Parametrul opţiunii `-name` (numele fişierului) trebuie inclus între ghilimele dacă el conţine caractere speciale (de tipul `*`).

- `grep` *text nume_fişier*

  Afişează toate liniile din *nume_fişier* care conţin *text*. Dacă textul conţine blancuri sau caractere speciale, el trebuie pus între ghilimele. Numele *nume_fişier* poate conţine caracterele * sau ?

  Exemple: `grep alias .bashrc`

  `grep "am fost" *doc`

1.6.6 Comparări de fişiere

- `diff` *nume_fis1 nume_fis2*

  Dacă fişierele sunt ASCII, afişează liniile care diferă în cele două fişiere.

- `sdiff` *nume_fis1 nume_fis2*

  Afişează conţinutul celor două fişiere, unul lângă altul, marcând cu | liniile care diferă şi cu `<` sau cu `>` liniile care există doar într-unul din cele două fişiere.

1.7 Indirectări

- *comanda1* | *comanda2*

  Execută comanda *comanda1* şi transmite rezultatul ca intrare pentru comanda *comanda2*.

  Exemple: `sdiff test1 test2 | more`

  Afişează rezultatul comenzii `sdiff` pagină cu pagină (conform comenzii `more`).

  `cat test* | grep "am fost"`

  Afişează acele linii din fişierele al căror nume începe cu test, care conţin caracterele ”am fost”.

- *comanda1* `>` *nume_fişier* Execută comanda *comanda1* şi scrie rezultatul în fişierul *nume_fişier* în loc să îl afişeze la consolă. Dacă fişierul există, el este rescris.

  Exemplu: `ls -l > dir`

- *comanda1* `>>` *nume_fişier*

  Execută comanda *comanda1* şi scrie rezultatul la sfârşitul fişierului *nume_fişier* în loc să îl afişeze la consolă. Dacă fişierul nu există, el este creat.

- *comanda1* `<` *nume_fişier*

  Execută comanda *comanda1*, preluând datele de intrare din fişierul *nume_fişier* în loc să le preia de la consolă.

1.8 Imprimarea fişierelor

- `lpr` *nume_fişier*

  Trimite la imprimantă conţinutul fişierului.

  Trebuie notat că această comandă este specifică sistemelor Unix de tip SVR4, în timp ce la sistemele de tip BSD ea are forma:

  `lp` *nume_fişier*

  Acesta este cel mai simplu mod să vă daţi seama ce tip de Unix utilizaţi!

  Observaţie: În reţeaua de calculatoare Pentium, această comandă **nu funcţionează** decât dacă fişierul este în format PostScript, acesta fiind singurul format recunoscut de imprimantă. De aceea, pentru a imprima un fişier text, daţi următoarele două comenzi:

  `a2ps` *nume_fişier* `> tmp.ps`

  `ghostview tmp.ps &`

  imprimare imprimaţi fişierul folosind butonul corespunzător al programului Ghostview. Înainte de a imprima un document mai lung, revedeţi regulamentul de funcţionare (anexa B).

- `lpq`

  Afişează coada de aşteptare a documentelor trimise la imprimantă de toţi utilizatorii.

1.9 Lansarea şi oprirea comenzilor

- *comandă* <↵ > – Lansează comanda în execuţie, în "foreground" (nu se mai pot da alte comenzi din fereastra în care a fost lansată, până la terminarea execuţiei comenzii).
- *comandă* & <↵ > – Lansează comanda în execuţie, în "background" (fără blocarea promptului).
- `<Ctrl> Z` – Suspendă execuţia comenzii curente, lansată în foreground.
- `bg` – Reporneşte, în background, o comandă suspendată cu `<Ctrl> Z`.
- `fg` – Trece în foreground o comandă lansată în background.
- `<Ctrl> C` sau `<Ctrl> D` – opreşte execuţia unei comenzi.

1.10 Comenzi pentru controlul proceselor şi utilizatorilor

- `who` – afişează lista utilizatorilor conectaţi în sistem

  `w` – afişează lista utilizatorilor conectaţi în sistem şi comanda pe care aceştia o execută;

  `whoami` – "Cine sunt eu?" – afişează numele de cont al utilizatorului;

  `finger` *nume* – "Cine este ...?" – afişează date despre toţi utilizatorii al căror nume de familie, prenume sau nume de utilizator este *nume* (printre altele, se afişează numele de utilizator, numele şi prenumele, precum şi ultima dată când şi-a citit poşta electronică);

- `ps` – lista proceselor.

  Sub Unix, mai multe comenzi sau programe pot fi rulate "în paralel" (de exemplu, dacă aveţi două ferestre deschise puteţi da comenzi care vor fi executate în mod independent). Fiecare comandă în curs de execuţie se numeşte *proces* şi este identificată printr-un număr, numit PID (Process IDentificator).

  Comanda `ps` fără nici un parametru afişează lista proceselor lansate de utilizatorul curent. Comanda poate fi lansată şi cu parametri: de exemplu, `ps -aux` listează toate procesele în curs de execuţie, cu informaţii despre utilizator, numărul procesului, timpul CPU ocupat, etc.

- `kill` *PID* – opreşte definitiv execuţia procesului cu numărul *PID*.

1.11 Ajutor!

Comanda care permite aflarea funcţiei şi sintaxei unei comenzi, precum şi multe alte informaţii despre comandă, este:

- `man` *nume_comandă*

  După lansarea ei, sunt active comenzile de deplasare descrise la `more` (paragraful 1.6.2).

- `man -k` *cuvânt_cheie* sau `apropos` *cuvânt_cheie*

  – caută cuvinte cheie în lista de comenzi

  `whatis` *cuvânt_cheie* – caută cuvinte cheie în lista de comenzi Unix şi afişează numai acele comenzi care conţin cuvântul ca atare (nu şi dacă el este parte a altui cuvânt mai lung).

  Pentru a observa diferenţa între `apropos` şi `whatis`, încercaţi, de exemplu,

  `apropos dos` şi

  `whatis dos`

1.12 Transferul fişierelor între DOS şi Unix

Fişierele **text** Unix se deosebesc de cele create sub DOS prin aceea că sfârşitul de linie este marcat doar prin line–feed şi nu prin combinaţia carriage return + line–feed ca în DOS. Pentru a transforma un format în altul se folosesc comenzile:

- `fromdos <` *nume_fis_DOS* `>` *nume_nou*
- `todos <` *nume_fis_Unix* `>` *nume_nou*

Pe diverse maşini, aceste comenzi se numesc diferit. Încercaţi `man -k dos`.

1.13 Diverse utilitare

- `date` – afişează data şi ora;
- `cal` – afişează calendarul lunii curente;

  `cal 08 1998` – afişează calendarul lunii august a anului 1998;
- `echo` *text* – afişează *text* pe ecran;

  `echo $VAR` – afişează valoarea variabilei de configurare `VAR` pe ecran;

- `xclock` – deschide o fereastră în care se afişează ora, în mod implicit în format analogic;

  `xclock -digital` – deschide o fereastră în care se afişează ora în format digital;

- `calc, xcalc` – emulează un calculator de buzunar;

- `sort` *nume_fişier* – sortează în ordine alfabetică liniile din fişier şi afişează rezultatul pe ecran;

- `clear` – şterge conţinutul ferestrei de lucru;

- `wc` *nume_fişier* – numără liniile, cuvintele şi caracterele din fişier şi afişează rezultatul pe ecran;

- `quota` – afişează spaţiul de disc ocupat de utilizator şi limita maximă alocată. Vă recomandăm să folosiţi această comandă şi să compactaţi fişierele pe care nu le utilizaţi în mod curent, pentru a nu depăşi limita impusă.

- `xterm` – deschide o nouă fereastră X.

  Poate avea diferiţi parametri, printre care:

  `-T` *nume_fer* sau `-title` *nume_fer* – setează numele ferestrei ca fiind *nume_fer* (implicit: XTerm)

  `-sb` – fereastră cu scroll–bar;

  `+sb` – fereastră fără scroll–bar;

  `-e` *comandă* – deschide fereastra şi execută în ea comanda;

  Exemplu:

  ```
  xterm -T Conex_Heineken -sb -e ssh heineken &
  ```

1.14 Exerciţii

EXERCIŢIUL 1

Iniţiaţi sesiunea de lucru.

Aflaţi care este numele sistemului de operare, numele maşinii pe care lucraţi şi numele sub care vă cunoaşte sistemul. Aflaţi ce alte informaţii deţine sistemul despre Dvs.

Introduceţi textul "EB-210" urmat de <↵> (Enter) la promptul din fereastra de lucru. Faceţi aceeaşi operaţiune folosind butoanele mouse–ului pentru a copia textul. Comentaţi utilitatea acestei ultime tehnici.

Cu excepţia acestui exerciţiu, NU IGNORAŢI NICIODATĂ MESAJELE DE EROARE afişate!!! (Ele sunt, la acest exerciţiu, inofensive.) Dacă nu ştiţi ce înseamnă, întrebaţi, dar NU LE IGNORAŢI!

EXERCIŢIUL 2

Deschideţi o nouă fereastră.

Folosind butonul din mijloc, schimbaţi-i dimensiunile. Mutaţi fereastra în colţul din stânga jos al ecranului.

Lansaţi în această fereastră primele patru utilitare prezentate în paragraful 1.13.

Închideţi fereastra nou creată.

EXERCIŢIUL 3

a) Deschideţi încă două ferestre. Observaţi efectul asupra căsuţei Desktop0.

b) Deplasaţi-vă din fereastră în fereastră (activând astfel una sau alta dintre ferestre), în două moduri:

- deplasând cursorul mouse–ului;
- folosind bara orizontală din partea de jos a ecranului.

c) Mutaţi o fereastră în Desktop1 astfel:

- mutaţi fereastra spre dreapta până când o jumătate a ei nu mai este vizibilă (observaţi efectul asupra căsuţei Desktop1);
- deplasaţi cursorul mouse–ului pe căsuţa Desktop1 şi apăsaţi butonul din stânga al mouse–ului;
- poziţionaţi fereastra în Desktop1.

Comentaţi utilitatea celor trei desktop–uri.

EXERCIŢIUL 4

a) Afişaţi lista fişierelor din directorul curent. Deschideţi o altă fereastră şi repetaţi comanda.

b) Afişaţi pe ecran, într-una din ferestrele de lucru, conţinutul fişierului `/usr/local/man/whatis`.

EXERCIŢIUL 5

a) Copiaţi fişierul `.cshrc` sub numele `test`. Listaţi conţinutul fişierului `test`.

Redenumiţi `test` în `test1`.

Listaţi conţinutul directorului curent. Ştergeţi fişierul `test1`.

b) Fără a le executa, încercaţi să prevedeţi efectul fiecăreia din următoarele instrucţiuni:

```
cp .bashrc temporar
mv temporar temp
rm temporar
rm temp
```

Executaţi aceste instrucţiuni şi observaţi efectul lor.

EXERCIŢIUL 6

Mutaţi-vă în directorul Dvs. *home*:

```
cd <↵>
```

Listaţi conţinutul acestui director.

Creaţi un nou director `metnum` în care veţi lucra de acum înainte la laboratorul de "Metode numerice în ingineria electrică". Listaţi noul conţinut al directorului *home*.

În directorul `metnum`, creaţi un nou director, `tema1`.

Aţi reuşit?

Vă recomandăm să creaţi câte un director pentru fiecare temă (`tema2, tema3, ...`) şi să lucraţi de fiecare dată în directorul corespunzător.

În directorul Dvs. home nu trebuie să existe nici un fişier (cu excepţia celor utilizate de sistemul de operare, al căror nume începe cu punct), ci numai directoare!

EXERCIŢIUL 7

Listaţi procesele în curs de execuţie. Mutaţi-vă în directorul Dvs. *home*. Într-o fereastră, daţi comanda:

```
more /usr/local/man/whatis
```

În altă fereastră, daţi comanda `ps`

Observaţi diferenţele faţă de rezultatul precedentei comenzi `ps`.

Lucrând în a doua fereastră, opriţi execuţia procesului care corespunde comenzii

`more usr/local/man/whatis`.

EXERCIŢIUL 8

a) Folosind comanda `man` aflaţi care este efectul comenzii `cal`.

b) Copiaţi fişierul `.bashrc` din directorul *home* în directorul `~/metnum/tema1/`.

Comparaţi efectul comenzilor

```
sort .bashrc
```

şi respectiv

```
sort .bashrc > sortat
```

(Indicaţie: încercaţi `ls` şi `more ...`).

EXERCIŢIUL 9

Comparaţi conţinutul fişierelor `~.bashrc` şi `~/metnum/tema1/.bashrc`, folosind `diff` şi `sdiff`. Efectuaţi aceeaşi operaţie asupra fişierelor `~.bashrc` şi `~/metnum/tema1/sortat`.

EXERCIŢIUL 10

Creaţi un fişier numit `dir` care să conţină lista fişierelor din directorul curent.

EXERCIŢIUL 11

Mutaţi-vă în directorul Dvs. home. Găsiţi (cu `find`) fişierul numit `sortat`.

EXERCIŢIUL 12

Blocaţi accesul la terminal cu comanda `xlock`.

Deblocaţi accesul şi încheiaţi sesiunea de lucru.

Intraţi din nou în sistem şi modificaţi-vă parola. Nu uitaţi noua parolă!

EXERCIŢIUL 13

Alcătuiţi o listă cu cele mai importante 10 comenzi Unix, în opinia Dvs. Aflaţi mai multe detalii despre aceste comenzi folosind comanda `man`.

Nu uitaţi să închideţi sesiunea de lucru înainte de a părăsi laboratorul.

Dacă vreţi să aflaţi mai multe despre Unix, vă recomandăm să consultaţi una din cărţile [55], [65] sau [73].

Capitolul 2

Editarea textelor
— `vi` **şi** `Emacs` —

Textul conceput ca un şir (liniar!) de caractere (litere, cifre, semne de punctuaţie şi caractere speciale, de exemplu <linie nouă>) reprezintă unul din conceptele fundamentale ale comunicării umane şi în particular a celei tehnico–ştiinţifice. El reprezintă baza majorităţii documentelor, fie că acestea sunt rapoarte ştiinţifice sau programe de calculator, reprezentate pe hârtie, pe ecran sau electronic (pe suport magnetic sau în memoria internă a calculatoarelor).

Pentru a facilita operaţiile de introducere în calculator, vizualizare şi modificare ale textelor s-au dezvoltat mai multe programe capabile să manipuleze interactiv texte, numite *editoare*.

Editarea textelor este o operaţie esenţialmente interactivă şi ea trebuie deosebită de cea de *procesare* a textelor, care este în principiu neinteractivă. Procesarea textelor presupune prelucrarea automată a textelor, bazată pe căutări şi substituiri condiţionate (ca de exemplu înlocuirea diacriticelor româneşti cu literele simplificate corespunzătoare, sau invers!). Cu toate că pentru procesare s-au dezvoltat programe specializate, cum sunt `sed`, `awk`, `lex`, etc. [36], un editor de texte performant trebuie să ofere şi el funcţii de procesare (cel puţin "*search and replace all*"). De asemenea, editarea textelor nu trebuie confundată cu *editarea documentelor,* în care conceptele de *font* şi aşezare în pagină sunt esenţiale.

Dacă sub sistemul de operare DOS editorul clasic este *WS* (*Word Star*), iar sub Windows editorul clasic este *MS-Word*, în cazul sistemului Unix editorul "nativ" este **vi**. Cu toate că se spune despre acest editor că este dificil de învăţat, cunoştinţele elementare (care presupun memorarea a 10 maxim 20 de mnemonici formate dintr-o literă) vă sunt suficiente pentru a vă putea descurca în orice versiune de sistem de operare Unix.

Un alt editor celebru pentru Unix, extrem de performant şi productiv, este **Emacs**. Şi în acest caz, cunoaşterea a 10-20 mnemonici (din păcate, altele decât în *vi*!) vă este suficientă pentru a putea efectua operaţii elementare de editarea textelor.

Acestea nu sunt singurele două soluţii posibile. În LMN sunt instalate şi alte editoare, mai puţin folosite, cum sunt *Joe* (editor ce emulează WS) şi *Xedit* (editor simplu de tip Note-pad).

Indiferent ce editor veţi folosi în viitor, vă recomandăm să vă familiarizaţi cel puţin cu două dintre acestea pentru a înţelege conceptele specifice editării de texte.

După ce v-aţi ales editorul, exersaţi cât mai mult cu el. Productivitatea Dvs. în continuare (indiferent la ce veţi folosi calculatorul în activitatea ştiinţifică şi tehnică) va depinde esenţial de viteza cu care reuşiţi să editaţi textele!

Scopul capitolului este de a deprinde operaţiile elementare de editare a textelor:

- lansarea editorului;
- introducerea unui text;
- modificarea unui text (ştergerea, adăugarea, copierea şi mutarea unor părţi din text, căutarea şi înlocuirea);
- citirea şi scrierea în fişiere;
- încheierea sesiunii.

2.1 Editarea textelor cu `vi`

Programul de editare a textelor `vi` [33] [52] a fost scris de *Bill Joy* când era student la Universitatea Berkley, ca o îmbunătăţire a editorului de linie `ed` dezvoltat de *Bell Labs* (AT&T), în cadrul proiectului UNIX.

`vi` este un editor puternic, care permite editarea interactivă a textelor. Editorul `vi` are un avantaj esenţial: este parte componentă a oricărui sistem Unix, indiferent de variantă. Dezavantajul principal al acestui editor constă în faptul că se bazează pe două moduri de lucru (text şi comandă), ceea ce îl face greu de acceptat de cei care au experienţa utilizării unui editor de documente (nu de texte!) de tip *WYSIWYG* (*What You See Is What You Get*) cum este MS-Word.

Varianta de `vi` instalată în laboratorul EB212 se numeşte `vim` (Vi IMproved) [27] şi conţine multe facilităţi suplimentare faţă de varianta clasică.

2.1.1 Apelare

- Lansarea variantei clasice:

  ```
  vi fis
  ```

  în care `fis` este opţional şi reprezintă numele fişierului ce trebuie editat.

- Lansarea variantei cu interfaţă grafică (GUI) în altă fereastră X:

  `vi -g fis`

  În acest caz se deschide o nouă fereastră (figura 2.1), în care sunt disponibile toate comenzile variantei clasice (şi multe altele, specifice **VIM**), în plus, o parte dintre operaţii putând fi efectuate folosind mouse–ul.

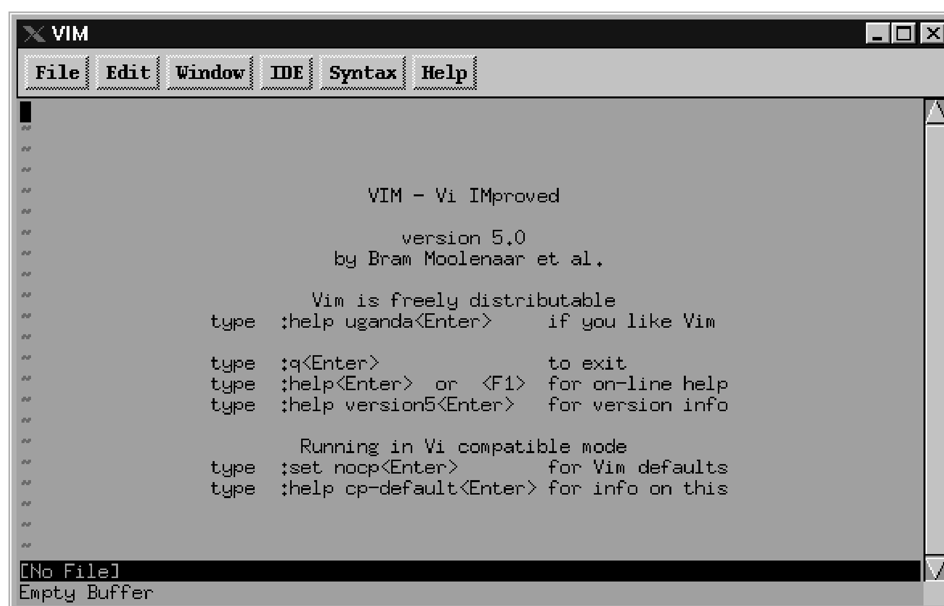


Fig. 2.1: Fereastra grafică a editorului VIM

Comenzile prezentate mai jos sunt cele de bază, disponibile în toate versiunile editorului `vi`.

2.1.2 Moduri de lucru

Există două moduri de lucru ale editorului `vi`:

- mod "comandă" (este cel iniţial, după lansarea în execuţie a editorului);
- mod "text" (modul în care introduceţi text, dar în care intraţi printr-o comandă).

Din modul de comandă se pot face următoarele operaţii:

- intrare în modul text;
- deplasarea cursorului;
- introducerea unor comenzi de editare;
- efectuarea unor căutari şi înlocuiri de şiruri de caractere;
- salvarea fişierului şi ieşirea din programul `vi`;

- altele (comenzi ex, Unix shell).

Din modul text se pot face următoarele operaţii:

- introducere text;
- trecere înapoi în mod comandă (cu <ESC>).

2.1.3 Modul comandă

Intrarea în mod text

Se poate face în mai multe variante:

a	- adăugare după cursor
A	- adăugare la sfârşit de linie
C	- modificarea restului liniei
i	- inserare înaintea cursorului
I	- inserare la începutul liniei
o	- linie nouă după linia curentă
O	- linie nouă înaintea liniei curente
R	- scriere "peste" (replace) – mai multe caractere
s	- înlocuire un caracter
S	- înlocuire întreaga linie

Comenzi de deplasare a cursorului

l	- dreapta
h	- stânga
j	- jos
k	- sus
<blanc>	- dreapta
w, W	- înainte cu un cuvânt
b, B	- înapoi cu un cuvânt
e	- sfârşit cuvânt
),(	- propoziţie următoare, curentă
}, {	- paragraf următor, curent
0	- începutul liniei curente
$	- sfârşitul liniei curente
^	- primul caracter diferit de blanc din linia curentă
+, -	- primul caracter al liniei următoare, precedente
<↵ >	- primul caracter al liniei următoare
H	- început ecran
M	- mijloc ecran

`L`	- ultima linie ecran
`Ctrl-F, Ctrl-B`	- un ecran înainte, înapoi
`Ctrl-D, Ctrl-U`	- jumătate de ecran în jos, în sus
`z` urmat de:	
<↵> . -	- linia cu cursorul plasată la început, mijloc, jos ecran
`Ctrl-L, Ctrl-R`	- redesenare ecran
`G`	- ultima linie din fişier
<n>G	- linia numărul <n> din fişier

Comenzi de editare

Au structura generală:

<operaţie><obiect>

Comenzi (operaţii) de bază:

`c`	- începutul unei modificări
`d`	- începutul unei ştergeri
`y`	- începutul unei copieri (yank)

Comenzile de editare acţionează asupra unui obiect. Acesta poate fi:

- cuvânt = toate caracterele până la un blanc sau un semn de punctuaţie
- propoziţie = toate caracterele până la un `.`, `!` sau `?`, urmate de un spaţiu
- paragraf = până la următoarea linie liberă
- secţiune = până la următorul titlu de secţiune, definit în mod corespunzător

Dacă obiectul este linia curentă, atunci numele obiectului este acelaşi cu numele operaţiei:

`cc`	- modifică linia curentă
`dd`	- şterge linia curentă
`yy`	- copiază linia curentă

Comanda precedată de un număr `n` acţionează asupra a `n` obiecte:

`2dd` - şterge două linii, începând cu cea curentă

Comenzi de editare

`cw`	- modificare cuvânt
`cc`	- modificare linie
`C`	- modificare de la cursor la sfârşit linie
`dd`	- ştergere linie
`D`	- ştergere rest linie
`dw`	- ştergere cuvânt

`d}`	- ştergere până la paragraf următor
`d^`	- ştergere înapoi până la început linie
`rx`	- înlocuire caracterul curent cu x
`u`	- undo ultima schimbare
`U`	- restaurare linie curentă
`x`	- ştergere caracter curent
`X`	- ştergere caracter la stânga
`.`	- repetă ultima modificare
`p`	- inserează ultimul text şters, după poziţia cursorului
`P`	- inserează ultimul text şters, înainte de poziţia cursorului
`:r <fis>`	- citeşte (include) fişierul <fis> în fişierul curent

Copiere şi mutare

`yy`	- copiere linie curentă
`"xyy`	- copiere linie curentă în bufferul x
`"xp`	- pune conţinutul bufferului x la poziţia cursorului

Diverse
`J` - uneşte două linii

Căutări şi înlocuiri

`/text`	- caută înainte text
`?text`	- caută înapoi text
`/,` sau `n`	- repetă ultima căutare, înainte
`N`	- repetă ultima căutare, înapoi
`:s /<text>/<text1>/`	- caută `<text>`, înlocuieşte cu `<text1>`
`:s /<text>/<text1>/g`	- caută `<text>`, înlocuieşte cu `<text1>`, în toate apariţiile lui `<text>` de pe linie
`1,$ s /<text>/<text1>/g`	- acelaşi lucru, de la linia 1 la ultima linie (notată `$`) din fişier

Salvare şi ieşire

`:x`	- ieşire, cu scriere dacă s-au făcut modificări
`:wq`	- scriere şi ieşire
`:w`	- scriere (salvarea fişierului curent)
`:w <fis>`	- salvarea fişierului curent sub numele <fis>
`:q`	- ieşire fără salvare
`:q!`	- ieşire fără salvare, chiar dacă s-au facut modificări
`:e <fis>`	- editarea fişierului `<fis>`

2.1.4 Setare opţiuni

- în mod comandă:

  `:set <opţiune>`

- în fişierul de configurare `.exrc`

  `set <opţiune>`

`<opţiune>` e de forma:

`<opt>`	- setare
`no<opt>`	- resetare
`x=val`	- setează x la valoarea val

Exemple:
`set ai` - autoindent
`set noai` - no autoindent
`set nu` - numere de linii afişate pe ecran
`set tabstop=8` - numărul de caractere pentru un TAB
`set wrapmargin=72` - marginea din dreapta la 72 de caractere

2.1.5 Câteva facilităţi ale interfeţei grafice cu utilizatorul

Interfaţa grafică a programului `vi` (în versiunea **VIM**) pune la dispoziţia utilizatorului o serie de comenzi selectabile cu ajutorul mouse-ului.

Comenzile sunt grupate în şase meniuri (vezi şi figura 2.1), după funcţiile lor: lucrul cu fişiere (**File**), comenzi de editare (**Edit**), lucrul cu ferestre (**Window**), facilităţi pentru scrierea de programe (**IDE**), definirea tipului de fişier, pentru a permite verificarea automată a sintaxei (**Syntax**) şi documentaţie (**Help**).

Meniurile din care pot fi selectate diferitele comenzi din interfaţa grafică cu utilizatorul sunt prezentate în figura 2.1.5.

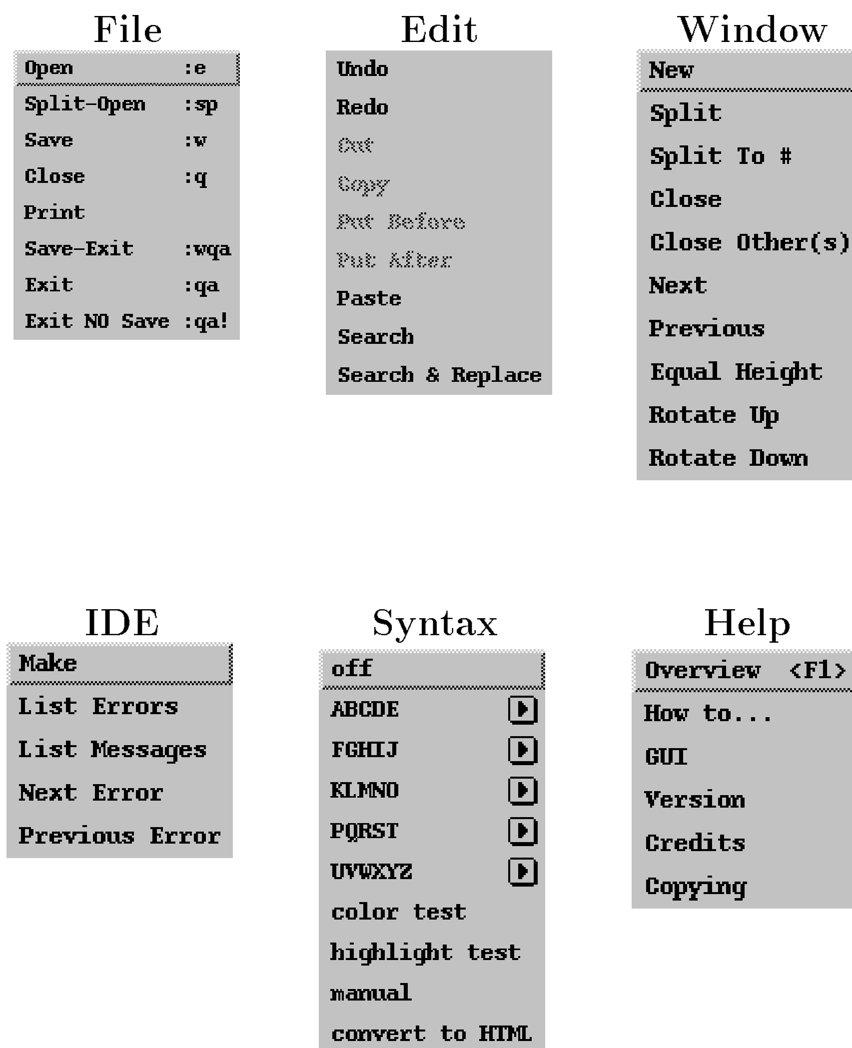


Fig. 2.2: Meniurile de comenzi ale interfeţei grafice cu utilizatorul a programului **VIM**

2.2 Editarea textelor cu `emacs`

GNU Emacs [75] este un program extrem de puternic ce permite realizarea unei game largi de operaţii de editare de texte, dezvoltare de programe sau management de fişiere. Emacs (Editor MACroS) a fost dezvoltat de Richard Stallman şi Guy Steele de la MIT şi plasat în domeniul licenţei gratuite GPL, domeniu gestionat de Free Software Foundation (înfiinţată tot de Stallman), ca o componentă a proiectului GNU. Reamintim că acest proiect urmăreşte dezvoltarea unui set de componente şi instrumente (software tools) care să fie alternative la sistemul UNIX clasic, dar să fie "free", adică oferite gratis.

Emacs este nu numai o excelentă alternativă la editorul **vi** ci este un adevărat mediu eficient de gestiune a documentelor (texte ale unor documente, programe, mesaje electronice, etc.).

2.2.1 Lansare şi ieşire din Emacs

- Lansare:

  `emacs`

  sau

  `emacs fisier`
- Pentru a ieşi din Emacs:

  ↑ x ↑ c

  sau

  ↑ u ↑ x ↑ c

  pentru a ieşi fără salvare.

2.2.2 Ecranul Emacs

Ecranul Emacs (fig. 2.3) este alcătuit din (de sus în jos):

- zona de meniuri (File, Edit, ...) – disponibilă numai la varianta X a programului;
- zona de butoane (Open, DirEd, Save, ...) – disponibilă numai la varianta X a programului;
- spaţiul de lucru (separat eventual în mai multe ferestre)
- linia de stare
- aria de ecou, în care sunt afişate mesajele Emacs
- opţional lista etichetelor cheilor de comandă (F1-F8)

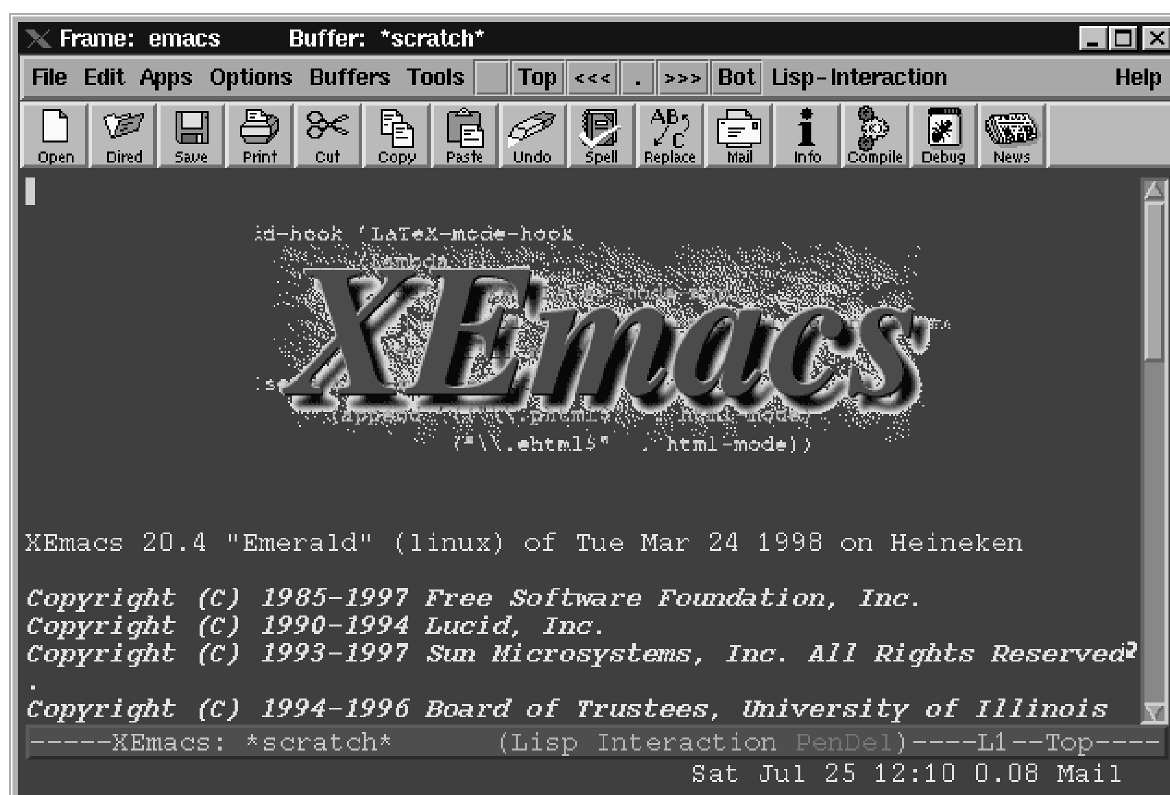


Fig. 2.3: Ecranul programului `emacs`

În orice ecran Emacs există un singur cursor la un moment dat, care marchează fereastra selectată.

Linia de stare conţine în mod tipic:

- Starea (`-----` – text nemodificat, `--***` – text modificat);
- Sistem (Emacs, RMAIL, etc.);
- Nume buffer (fişier);
- Tipul şi încărcarea maşinii (%);
- Mail – mesaj electonic necitit;
- Modul: Text, C, FORTRAN, etc. ;
- Poziţia ferestrei în fişier.

Pentru a transmite comenzi în Emacs se folosesc două chei de comandă:

<Control>	simbolizată ↑
<Meta>	simolizată ↓

apăsate simultan cu caracterul următor. Tasta <Meta> este mapată pe tasta <Alt> sau pe <Esc> (caz în care nu trebuie apăsate simultan).

2.2.3 Comenzi de deplasare a cursorului

↑ b	- înapoi (back)
↑ f	- înainte (forward)
↑ p	- sus (previous)
↑ n	- jos (next)
↑ a	- început linie (mnemonica "alta")
↑ e	- sfârşit linie (end)
↓ b	- înapoi cu un cuvânt
↓ f	- înainte cu un cuvânt
↓ m	- la primul caracter neblanc
↓ a	- propoziţie înainte
↓ e	- propoziţie înapoi
↓]	- paragraf înainte
↓ [	- paragraf înapoi
↓ <	- început fişier
↓ >	- sfârşit fişier
↑ v	- scrie înainte (în jos v)
↓ v	- scrie înapoi (invers v)
↑ x <	- scroll la stânga
↑ x >	- scroll la dreapta
↑ x]	- pagină înainte
↑ x [	- pagină înapoi

2.2.4 Comenzi de ştergere

↑ d	- şterge caracter (delete)
↓ d	- şterge cuvânt
<Del>	- şterge caracter anterior
↑ k	- şterge până la sfârşit de linie (kill)
↓ k	- şterge propoziţie
↑ x ↑ k ↑ k	- şterge linia curentă (macrocomandă)
↑ x ↑ o	- şterge linie albă
↑ x u	- anulează ştergerea (extended undelete)

2.2.5 Comenzi de formatare–indentare

↑ o	- inserare linie albă
↓ s	- centrare linie
↓ i	- indent
↑ ↓	- indent la acelaşi nivel
↑ ↓ o	- rupe linie şi indentează
↑ ^	- uneşte linii indentate
↑ x f	- setează marginea din dreapta pe poziţia cursorului
↑ u N ↑ x f	- setează marginea din dreapta la coloana N

2.2.6 Comenzi pentru blocuri

↑ <Spc>	- setează marcare bloc
↑ x↑ x	- arată bloc
↑ w	- şterge bloc ("cut")
↓ w	- memorează bloc (pentru mutare)
↑ y	- inserează copie bloc ("yank"-paste)
↑ x↑ u	- înlocuieşte litere din bloc cu majuscule
↑ x↑ l	- înlocuieşte litere din bloc cu minuscule

2.2.7 Comenzi pentru fişiere (încărcare-salvare)

↑ x↑ s	- salvează fişier (extended save)
↑ x↑ w	- scrie fişier cu redenumire (extended write)
↑ x↑ f	- încarcă fişier (extended find)

2.2.8 Căutare şi înlocuire

↑ s text ↑ s	- căutare înainte (search)
↑ r text ↑ r	- căutare înapoi
<Esc>	- ieşire din căutare
↑ g	- abandon căutare
↑ %	- căutare şi înlocuire cu următoarele subcomenzi:

<Spc>	- înlocuieşte şi continuă
<Del>	- nu înlocui şi continuă
,	- înlocuieşte şi opreşte
!	- înlocuieşte toate apariţiile neinteractive

2.2.9 Comenzi pentru ferestre

↑ x2	- împarte fereastra vertical
↑ x5	- împarte fereastra orizontal
↑ x0	- elimină fereastra curentă
↑ x1	- elimină celelalte ferestre
↑ xo	- mută cursor în altă fereastra

2.2.10 Lucrul cu macrocomenzi

↑ x (....↑ x)	- definiţie macrocomandă
↑ x e	- lansează ultima macrocomandă definită
↑ u ↑ x (....↑ x)	- adaugă noi comenzi la definiţia ultimei macrocomenzi

2.2.11 Comenzi pentru dezvoltarea programelor

↑ ↓ a, ↑ ↓ e	- deplasează cursor la început/sfârşit funcţie
↑ ↓ f, ↑ ↓ b, ↑ ↓ p, ↑ ↓ n	- deplasează cursor în cadrul unei expresii cu paranteze
↑ ↓ h	- defineşte bloc peste funcţie
↑ j	- sfârşit linie cu indentare
↑ ↓	- reindentează liniile unui bloc
↑ q <Tab>	- inserează Tab
↑ (	- inserează pereche de paranteze
↓ ;	- adaugă comentariu
↑ ↓ j	- sfârşit comentariu
↑ u - ↑ x	- şterge comentariu din linia sau blocul curent
↓ x compile	- compilează program
↑ x`	- mergi la următoarea linie a erorii de compilare

2.2.12 Diverse

↑ g	- abandonează comandă
↑ h	- acces la help
↑ ht	- execută tutorial "on-line" (ieşire cu ↑ x k <↵ >)
↓ $	- spelling cuvânt
↓ x	- spelling fişier
↓ !	- execută comandă UNIX
↑ u ↓ !	- execută comandă UNIX cu inserarea răspunsului în fişier
↓ x tex-mode	- invocare mod TEX
↓ x latex-mode	- invocare mod LATEX
↑ xd	- invocare editor de directoare pentru managementul fişierelor (extended DirEd), după care sunt disponibile comenzile: c = copiere, f = încărcare, dx = ştergere fişier

2.3 Exerciţii

EXERCIŢIUL 1

Generaţi un fişier cu numele `vi.txt` (dacă folosiţi editorul `vi`), respectiv `emacs.txt` (dacă folosiţi editorul `emacs`) care conţine descrierea comenzilor editorului folosit pe care le consideraţi indispensabile (nu mai mult de 10).

EXERCIŢIUL 2

Generaţi un fişier cu numele `cv.form` care conţine formularul unui Curriculum vitae (Nume, Data şi locul naşterii, Adresă, Ocupaţie actuală, Studii, Realizări, Preocupări, ...)

EXERCIŢIUL 3

Folosind fişierul `cv.form` generaţi propriul Dvs. CV în fişierul `cv.txt`.

EXERCIŢIUL 4

În directorul ~`/tema2/`, generaţi un fişier cu numele `referat.form` care conţine formularul unui referat de laborator (conform modelului de la pagina 112).

Fişierul va avea conţine următoarele informaţii:

`Antet:` Universitatea "Politehnica" din Bucureşti – Laboratorul de metode numerice
`Titlu:` Cuvântul `Titlu`
`Autor:` Numele Dvs.
`Adresa de posta electronica, grupa, facultate`
`Instructor:` Numele instructorului

Următoarele rânduri se introduc ca atare:

```
Introducere
Continut
Concluzii
Lista bibliografica
Anexe
```

Notă: adresa de poştă electronică se obţine concatenând numele Dvs. de utilizator (obţinut cu comanda `whoami`) cu şirul de caractere `@lmn.pub.ro`.

EXERCIŢIUL 5

Scrieţi un fişier ~`/tema1/unix.txt` ce conţine descrierea principalelor comenzi Unix. Inseraţi în acest text părţi din descrierea `man` a comenzilor selectate, folosind funcţia de

marcare şi copiere între ferestre.

EXERCIŢIUL 6

Scrieţi, în directorul ~`/tema1/`, un script Unix (par. 1.5) numit `azi` care:

- afişează textul "Hello" urmat de numele Dvs. de utilizator;
- Afişează data şi ora;
- Afişează spaţiul de disc ocupat şi cel maxim alocat Dvs.;
- Afişează numele maşinii şi shell-ul folosit.

Executaţi acest script prin invocarea numelui său. (Dacă nu reuşiţi, revedeţi şi paragraful 1.6.4).

Capitolul 3

Poşta electronică
— `elm` —

Poşta electronică (*electronic mail* sau pe scurt *e-mail*) este unul din cele mai importante servicii oferite de Unix. Capacitatea de a trimite mesaje la indivizi sau grupuri de persoane prin intermediul reţelei de calculatoare aparţine tradiţional Unix-ului, dar ulterior a fost preluată şi de alte sisteme de operare (de exemplu Windows).

Poşta electronică (E) este o alternativă la serviciile tradiţionale de poştă (P) sau telefon (T), dar cu trăsături specifice: ea este asincronă (ca serviciul P), are o viteză de transmitere mai mică decât T, dar incomparabil mai mare decât P, este un intermediar între *scripta manent* (P) şi *verba volant* (T), în schimb are două avantaje majore faţă de serviciul telefonic: preţul mic şi capacitatea de a comunica simultan cu grupuri mari.

Principalele **funcţii** oferite de serviciul e-mail sunt:

- schimbul de mesaje între cunoscuţi;
- transmiterea de documente în format electronic;
- abonamente la periodice electronice;
- transmiterea informaţiilor care se pot reprezenta digital (programe executabile, sunete, grafică, animaţie, secvenţe video, etc.).

Aceste funcţii şi caracteristici detemină un nou mod de colaborare profesională în care rolul hârtiei se diminuează (*paperless office*) [73].

Tendinţa de globalizare a comunicaţiilor digitale a condus la conceptul de *ciberspaţiu*, care reuneşte mulţimea tuturor informaţiilor reprezentate în calculatoarele conectate în reţeaua mondială. Actorii ciberspaţiului sunt utilizatorii acestor calculatoare, fiecare identificat printr-o adresă de e-mail. Persoanele fără o astfel de adresă sunt echivalentul modern al celor care în viziunea tradiţională nu au adresă poştală ("homeless") sau nu au telefon. O adresă de e-mail are următoarea structură tipică [50]:

`Nume@Sistem.Subdomeniu.Domeniu`

în care:

- `Nume` este numele utilizatorului;
- `Sistem` este numele calculatorului sau reţelei locale de calculatoare;
- `Subdomeniu` este numele reţelei de calculatoare de nivel superior;
- `Domeniu` este domeniul naţional căruia îi aparţine calculatorul, conform codificării:

`at` = Austria
`au` = Australia
`ch` = Elveţia
`de` = Germania
`es` = Spania
`fr` = Franţa
`gr` = Grecia
`jp` = Japonia
`ro` = România
`uk` = Marea Britanie

În cazul Statelor Unite, domeniile sunt:

`com` = comercial

`edu` = educaţional

`gov` = guvernamental

`mil` = militar

`org` = organizaţii nonprofit

`net` = organizaţii ce gestionează reţele de calculatoare

`int` = organizaţii internaţionale

Cât timp aveţi un cont deschis în LMN, adresa Dvs. de e-mail este

`Nume@lmn.pub.ro`

`pub` fiind codul subdomeniului Universitatea "Politehnica" din Bucureşti. Adresele e-mail nu sunt "case–sensitive", dar pentru siguranţă preferaţi literele mici.

Pentru a beneficia de serviciile de poştă electronică, trebuie să aveţi propria adresă e-mail şi să cunoaşteţi utilizarea unui program de e-mail. Tradiţionalul program Unix de e-mail este `mail` [50], dar în versiunile recente s-au dezvoltat programe noi: `mailx` (SVR4) şi `Mail` (BSD), virtual echivalente.

În prezent sunt populare programele `elm` [37] (mai uşor de învăţat), Pine [18] (cu funcţii puternice de transmitere a fişierelor binare), sau componentele e-mail din NetScape [15] sau Emacs [75].

Conceptele specifice poştei electronice

Un mesaj electronic este alcătuit din textul propriu-zis al mesajului precedat de un antet cu structura:

`Date`: data transmiterii

`From`: adresă expeditor

`To`: adresă destinatar

`Subject`: subiectul mesajului (o linie)

`Cc`: lista destinatarilor cărora li se transmit copii ale mesajului ("carbon copy").

Fiecare mesaj are un identificator unic, iar dacă destinatarul nu a fost găsit, mesajul se întoarce la expeditor.

Mulţimea mesajelor primite se colectează într-o *căsuţă poştală* a utilizatorului. După citire ele pot fi lăsate în căsuţă (nerecomandat!), şterse, salvate într-un fişier, avansate altui destinatar (*forward*) sau mutate în altă căsuţă poştală a aceluiaşi utilizator (clasificate).

În corespondenţa electronică se recomandă respectarea următoarelor **reguli de stil** [69]:

- folosiţi mesaje scurte, cu paragrafe de max. 15 linii;
- lungimea liniei de max. 65 caractere;
- separaţi paragrafele cu linii albe;
- începeţi mesajul cu numele real al destinatarului şi încheiaţi cu numele Dvs. real;
- respectaţi regulile gramaticale în folosirea majusculelor;
- nu "bombardaţi" cu mesaje persoane cunoscute, dar mai ales necunoscute;

- folosiţi un stil de scriere decent şi politicos;
- nu utilizaţi caractere de control;
- când daţi replici la un mesaj, includeţi şi textul receptat precedat de caracterul >, după modelul:

  `> Ne putem vedea astăzi?`

  `Da, la ora 14:30.`

- în încheierea mesajului puteţi adăuga o "semnătură electronică", după modelul

  `Nume, Poziţie`

  `Adresă,`

  `Tel/Fax`

  Unii expeditori mai adaugă şi o deviză personală sau un motto. Vă recomandăm să păstraţi concizia şi eleganţa acestei semnături.

- în mesajele electronice se folosesc şi elemente non-verbale ("smileys"):

  `:-)` zâmbet (după glume);

  `:-(` supărare;

  `:-o` mirare;

  `;-)` aluzie;

  `:*` sărutări!

  Vă puteţi imagina că aceste semne reprezintă o faţă umană rotită cu 90°.

Obiectivele capitolului constau în:

- înţelegerea conceptelor specifice poştei electronice;
- căpătarea deprinderilor esenţiale necesare utilizării poştei electronice: citirea mesajelor, transmiterea mesajelor, gestionarea listelor destinatarilor (aliasuri), gestionarea mesajelor primite (ştergerea, mutarea, salvarea).

În continuare se prezintă pe scurt programul **elm**, dezvoltat de Dave Taylor, care are avantajul principal că este distribuit gratuit [1].

3.1 Apelare

`elm <↵>`

Programul de poştă electronică `elm` oferă facilităţi pentru:

- citire a mesajelor primite
- trimitere de mesaje
- menţinerea listei de "alias"-uri personale
- ştergerea mesajelor primite sau mutarea lor în alte căsuţe poştale
- modificarea opţiunilor programului `elm`

Pentru documentare ("help") se dă comanda: `?` urmat de:

`<nume comandă>` – afişează informaţii despre această comandă
`?` – afişează lista tuturor comenzilor disponibile

3.2 Structura ecranului `elm`

Ecranul tipic al programului `elm` este prezentat în figura 3.1. El conţine următoarele zone:

- Linia de informaţii – în care sunt afişate numele de fişier al căsuţei poştale principale şi numărul de mesaje pe care aceasta le conţine;
- Lista de mesaje primite;
- Zona de help;
- Linia de comenzi.

3.3 Citirea mesajelor primite

Se selectează mesajul dorit cu una din comenzile următoare (implicit este selectat ultimul mesaj primit):

`k` sau ↑ - selectarea precedentului mesaj neşters
`j` sau ↓ - selectarea următorului mesaj neşters
`K` - selectarea mesajului anterior
`J` - selectarea mesajului următor
`n` - selectarea mesajului `n`

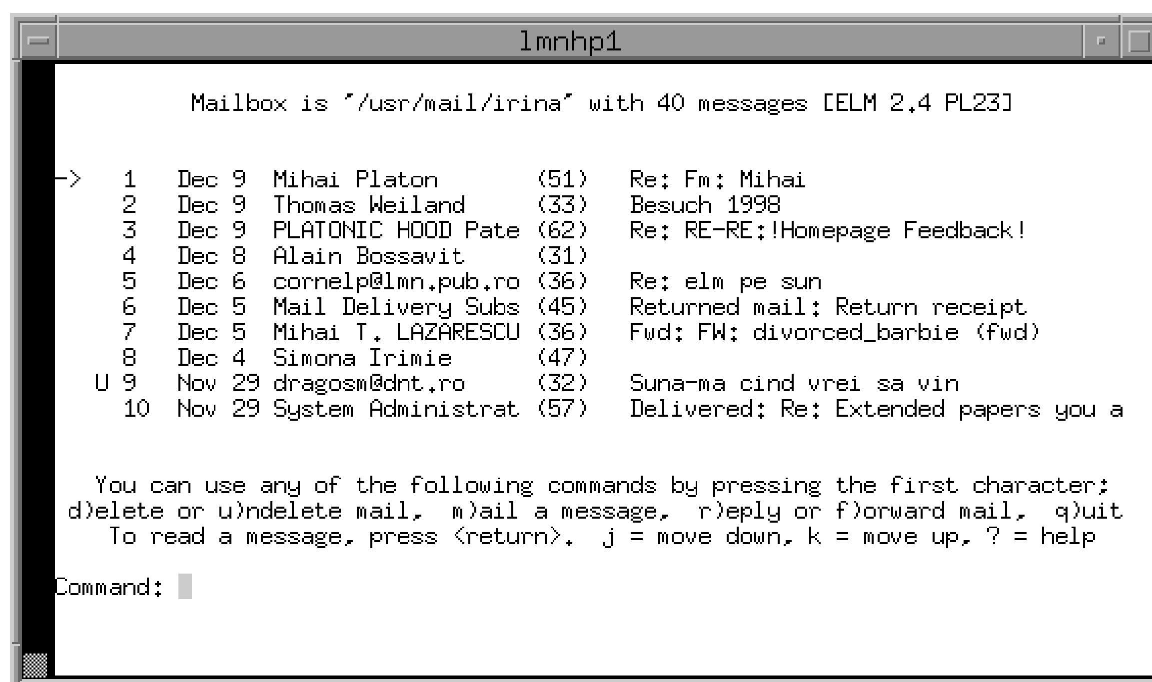


Fig. 3.1: Ecranul programului `elm`

`+` - avans un ecran

`-` - înapoi un ecran

Pentru citirea mesajului selectat:

`<↵ >` - citirea mesajului curent fără a afişa antetul (header-ul) mesajului

`h` - citirea mesajului curent, cu afişarea antetului (header)

În cursul citirii unui mesaj:

`<blanc>` - avansează un ecran

`<↵ >` - avansează o linie

↑ - mesaj precedent

↓ - mesaj următor

3.4 Trimitere de mesaje

Se introduce una din comenzile:

`m` - trimitere mesaj (mail)

`r` - răspuns expeditorului mesajului curent (reply)

`g` - răspuns expeditorului şi tuturor recipienţilor mesajului curent (global)

`f` - avansează mesajul curent unui alt destinatar (forward)

`b` - avansează mesajul curent unui alt destinatar, ca şi cum ar fi fost trimis de expeditorul iniţial (bounce)

Programul solicită:

- adresantul mesajului – listă de adrese sau aliasuri separate prin blancuri (fac excepţie comenzile `f` şi `g`)
- subiect
- cc (carbon copy) - listă de adrese sau aliasuri, separate prin blancuri

Se intră apoi într-un editor de text, în mod implicit `vi`. După comanda de scriere şi salvare a mesajului, utilizatorul poate:

`s` - să trimită mesajul (send);

`e` - să reintre în editarea mesajului (edit);

`h` - să editeze header-ul mesajului (headers);

`f` - să renunţe la trimiterea mesajului (forget-it).

Pentru crearea unor headere personalizate, se foloseşte fişierul

`$HOME/.elm/elmheaders`

Printre altele, în el se poate introduce linia:

`Return-Receipt-To:` *user*`@lmn.pub.ro`

utilă pentru a avea confirmarea de primire a mesajelor trimise.

3.5 Pseudonime (aliasuri)

Se dă comanda:

`a`

În modul "alias" sunt disponibile comenzile:

`n` - creare alias nou (new)

`a` - creare alias din adresa expeditorului mesajului curent

Se solicită:

- alias (nume scurt)
- nume de familie (last name)
- prenume (first name)
- comentariu opţional
- adresă de e-mail (sau listă de adrese, pentru a crea un grup)

? - help pentru comenzile disponible în “alias”

d - ştergerea unui alias (delete)

c - modificare alias curent (change)

<↵ > - verificarea adresei de e-mail pentru alias-ul curent

3.6 Ştergerea / mutarea mesajelor în altă căsuţă poştală

d - şterge mesaj curent (delete)

s - salvează mesaj curent în altă căsuţă poştală şi îl marchează ca şters

C - salvează mesaj curent în altă casuţă poştală fără să îl marcheze ca şters

c - schimbarea căsuţei poştale (change)

- fişier fis în directorul `$HOME`: ~`/fis`
- fişier fis în directorul Mail: `=/fis`
- fişierul de mesaje primite: `<`
- fişierul de mesaje trimise: `>`
- casuţă postală “principală”: `!`

Sunt utile comenzile:

t - marcare mesaj curent pentru operaţii ulterioare: salvare, ştergere, etc. (tag)

Ctrl-t - marcarea tuturor mesajelor care îndeplinesc o condiţie, în vederea unor operaţii ulterioare

3.7 Modificarea opţiunilor

- fiind în mod comandă se introduce:
  - o

  Sunt disponibile operaţiile:
  - Schimbarea nivelului de utilizator:

    u – 0 - începător; 1 - familiar; 2 - expert
  - Afişare mail:

    d – implicit `builtin` sau `usr/bin/more`
  - Editor:

    e – implicit `/usr/bin/vi`

– Folder (unde se ţin mesajele primite):

  f – implicit `$HOME/mbox`

– Criteriul de sortare a mesajelor:

  s - implicit `received` (în ordinea primirii, cel mai recent primul)
  Se poate folosi comanda

  R – sortare în ordine inversă, implicit `reverse received` (în ordinea primirii, cel mai recent ultimul)

– Căsuţa unde se salvează copiile mesajelor trimise:

  o – implicit `$HOME/mbox`
  Se recomandă folosirea unei căsute poştale dedicate, alta decât `mbox`, de exemplu `$HOME/Mail/outbox`. Pentru ca opţiunea să devină activă trebuie definit `copy = ON` în fişierul `elmrc`.

– Programul de imprimare:

  p – implicit `lp` sau `lpr`
  Cum imprimanta din laboratorul EB212 nu funcţionează în mod text, modificaţi această opţiune astfel: `/bin/cat %s | a2ps -c | lpr`

– Numele utilizatorului:

  y

– Forma cursorului:

  a – OFF (bară video-invers); ON (săgeată)

– Afişare numai numele persoanei în alias (Names Only):

  n – OFF (afişează şi adresa de Internet); ON (afişează numai numele)

• în fişierul ~/`.elm/elmrc`:

  Fişierul conţine variabile grupate pe următoarele categorii:

  – formatarea afişărilor
  – dispozitive ("devices") utilizate
  – directoare şi fişiere de mesaje
  – opţiuni/răspunsuri implicite

Toate opţiunile sunt date prin setarea unor variabile. Instrucţiunile sunt de forma: <variabilă> = <valoare>
Exemplu:

```
copy = ON

askcc = ON

editor = /usr/bin/vi

names = ON
```

3.8 Ieşirea din elm

x - ieşire fără modificarea căsuţei poştale

q - ieşire cu modificarea căsuţei poştale

3.9 Exerciţii

EXERCIŢIUL 1

Folosind editorul de texte preferat, generaţi un fişier numit logo care să conţină numele Dvs., poziţia (student în grupa ... la facultatea ...), adresa poştală, telefon, adresa electronică. Acest fişier poate să conţină şi o deviză personală (motto) dar trebuie să fie formatat cât mai elegant (încadrat între două linii de marcaj alcătuite din semne speciale: *, = sau -).

Mutaţi fişierul logo în directorul ~/.elm

EXERCIŢIUL 2

Lansaţi elm, intraţi în modificare opţiuni şi setaţi ordinea mesajelor astfel încât cel mai recent primit mesaj să apară primul în listă. Salvaţi opţiunile.

EXERCIŢIUL 3

Editaţi fişierul ~/.elm/elmrc, pentru a seta salvarea mesajelor trimise şi fişierul de "semnătură" (localsignature) ca fiind ~/.elm/logo (conform paragrafului 3.7).

EXERCIŢIUL 4

Scopul acestui exerciţiu este de a vă permite să creaţi, evitând greşelile de tastare, o listă de adrese e-mail ale colegilor de grupă.

Trimiteţi un scurt mesaj electronic colegului de grupă care vă urmează în lista alfabetică a numelor (ultimul de pe listă va trimite un mesaj primului).

Avansaţi (forward) copia tuturor mesajelor primite, colegului următor.

Salvaţi toate mesajele primite într-o nouă căsuţă poştală, numită grupa, plasată în directorul ~Mail (paragraful 3.6).

Constituiţi un alias numit grupa care să conţină adresele colegilor de grupă. Folosiţi facilitatea "cut and paste" (cu ajutorul mouse–ului) pentru a nu risca să introduceţi eronat adresa.

Ştergeţi mesajele din căsuţa poştală `grupa`.

EXERCIŢIUL 5

Expediaţi o copie a fişierului `cv.txt` realizat în tema precedentă (propriul Dvs. CV) către toţi colegii de grupă.

Creaţi un nou fişier care să conţină toate CV–urile primite. Editaţi acest fişier astfel încât să conţină doar lista numelor studenţilor şi data naşterii.

Trimiteţi acest mesaj colegului următor, şi în copie (cc) instructorului. (*À propos: cum faceţi să aflaţi adresa de poştă electronică a instructorului Dvs.?*)

Trimiteţi un mesaj de felicitare colegilor născuţi în luna curentă.

EXERCIŢIUL 6

Trimiteţi un mesaj instructorului, în care explicaţi funcţionarea comenzii Unix de imprimare a unui mesaj electronic, propusă în paragraful 3.7.

Ştergeţi toate mesajele primite din căsuţa poştală principală. Mutaţi-vă în căsuţa poştală în care sunt salvate automat mesajele trimise de Dvs. şi ştergeţi primul scurt mesaj trimis (exerciţiul 4).

Capitolul 4

Editarea documentelor — LaTeX —

Rezultatul principal al activităţilor de cercetare îl constituie realizarea unor *documente tipărite*: cărţi, articole în reviste de specialitate sau rapoarte ştiinţifice. Spre beneficiul întregii omeniri şi, implicit, al oamenilor de ştiinţă, de cinci sute de ani, de la apariţia tiparului, cunoştinţele tipografice s-au perfecţionat continuu, devenind o adevărată artă.

Spre deosebire de textele pure, *documentele* tipărite sunt caracterizate şi prin fonturile folosite (tipurile şi corpurile de literă) precum şi de aşezarea în pagină (inclusiv plasarea imaginilor şi eventual a formulelor matematice şi chimice).

Apariţia calculatoarelor a avut impact major şi asupra acestei activităţi. Realizarea cu calculatorul a unor documente tehnice şi ştiinţifice de calitate, în vederea multiplicării lor se face în prezent în două moduri principial diferite:

- **într-un pas**, cu editoarele de documente (*word processing*) gen MS Word, la care efectul editării este vizibil imediat pe ecran (WYSIWYG - "What You See Is What You Get) şi
- **în doi paşi**, folosind mai întâi un editor de texte pure (**vi**, **Emacs**, etc.) în care pe lânga textul documentului se adaugă specificaţiile de formatare (folosind convenţiile unui limbaj de tip "mark-up" care permite adăugarea indicaţiilor pentru "tipograf") şi apoi un program neinteractiv de formatare, care interpretează fişierul text de descriere a documentului şi produce un fişier de ieşire (codificat corespunzător, de obicei în format binar) reprezentabil pe ecran sau la imprimantă.

Aparent, prima metodă este ideală, dar se constată că în majoritatea cazurilor rezultatele obţinute cu ea nu satisfac standardele profesionale. Se poate spune că primul mod este echivalentul modern al utilizării maşinii clasice de scris, pe când al doilea corespunde maşinii tipografice de cules.

În anul 1977, Donald Knuth, profesor la Universitatea Stanford şi unul din cei mai mari specialişti în calculatoare din toate timpurile, a început lucrul la un proiect intitulat

TEX (se pronunţă *tehi*), care urmărea realizarea unui sistem de procesare a textelor care să permită realizarea unor "cărţi frumoase", conţinând inclusiv formule matematice complexe [53]. Evident, el a ales al doilea mod de abordare (în doi paşi) şi în plus a definit un standard (*DVI = DeVice Independent*) pentru formatul fişierului de ieşire. Sistemul TEX de formatare a documentelor, realizat de Knuth satisface cele mai exigente standarde privind:

- **calitatea** (absolut remarcabilă a) rezultatului;
- **portabilitatea** (poate rula pe diferite platforme UNIX, VMS, Windows, iar fişierele DVI pot fi reprezentate pe cele mai diverse dispozitive: ecrane, imprimante matriciale, imprimante cu laser, etc.);
- **disponibilitatea** (în ciuda efortului imens pentru realizarea lui, pachetul de software TEX este disponibil fără plata licenţei şi distribuit de o organizaţie nonprofit numită TUG – TeX Users Group [24]);
- **stabilitatea** (Donald Knuth a anunţat "îngheţarea" efortului de dezvoltare pentru a oferi stabilitate produsului).

Toate aceste calităţi au determinat succesul enorm al proiectului şi l-a făcut pe Gordon Bell să afirme că "*TEXeste cea mai importantă invenţie a secolului în domeniul tipografiei şi poate fi pusă pe acelaşi nivel cu invenţia tiparului de către Gutenberg*".

Complexitatea şi mai ales flexibilitatea sistemului TEX îl fac totuşi relativ greu de folosit. Din acest motiv Leslie Lamport a început în 1985 dezvoltarea unui sistem numit LATEX (LAyout TeX, pronunţat *leitehi*), bazat pe TEX, dar care să permită utilizatorului (de obicei autorul documentului) să se poată concentra asupra structurii şi organizării logice a documentului şi nu pe detaliile tipografice.

LATEX este un standard "de facto" pentru editarea documentelor ştiinţifice şi tehnice de înaltă calitate (articole pentru publicaţii ştiinţifice, rapoarte tehnice, cărţi, prezentări, dar şi scrisori, broşuri sau lucrări umaniste) [63].

Spre deosebire de TEX, care se poate considera ca un nivel de bază (*low–level*), LATEX este un sistem de nivel înalt (*high–level*) şi permite:

- structurarea documentelor mari;
- includerea de tabele, figuri, formule matematice complexe;
- cuprins, liste bibligrafice, index generate automat.

Se poate spune că TEX este "zeţarul" iar LATEX este "tehnoredactorul" [71].

Sistemul LATEX realizează formatarea textelor conform celor mai exigente standarde fără să fie necesare prea multe eforturi din partea utilizatorului. Este adevărat că aspectul

general al documentului realizat cu LaTeX se modifică greu însă rezultatul este profesional. Toate marile reviste de specialitate au definit stiluri LaTeX obligatorii pentru autorii articolelor [25].

Proiectul LaTeX a respectat toate caracteristicile TeX, rezultatul fiind disponibil gratis prin reţeaua de calculatoare [26] sau la preţ nominal, pe CD ROM; în plus el este relativ uşor de învăţat şi utilizat.

Spre deosebire de TeX, dezvoltarea LaTeX nu este îngheţată: ea a fost preluată de proiectul **LaTeX3**, urmărindu-se dezvoltarea comenzilor existente (stil, tratare erori, interfaţă) dar şi implementarea conceptului de *hipertext*. Aceasta va face posibilă utilizarea în viitor a sistemului şi pentru generarea documentelor electronice "on-line" (cele care satisfac standardele SGML-HTML şi PDF).

Obiectivele capitolului de faţă constau în:

- familiarizarea cu conceptele tehnoredactării profesionale (stil, elemente de structură, obiecte – inclusiv cele matematice, macrodefiniţii, structura paginii, fonturi);
- însuşirea deprinderilor necesare editării rapoartelor profesionale (inclusiv a rapoartelor de laborator LMN) şi a articolelor în reviste internaţionale (IEEE).

4.1 Etapele utilizării LaTeX

Actualmente, LaTeX este instalat în LMN pe maşinile: lmnsun, lmnhp5 şi pe toate maşinile Linux. Există două versiuni instalate: versiunea 2.09 (lmnsun) şi respectiv 2ε (pe maşina lmnhp5 şi pe maşinile Linux).

Un model de fişier LaTeX cuprinzând definiţii utile este fişierul `latex_head.tex`, al cărui conţinut este listat în paragraful 4.10.1.

Etape:

1. Se scrie fişierul cuprinzând documentul, cu extensia .tex. Să presupunem că numele fişierului este `fis.tex`
2. Se dă comanda de procesare cu LaTeX a fişierului:

   ```
   latex fis
   ```

   Fişierul creat în urma acestei comenzi este `fis.dvi`.
   Uneori, pentru a obţine rezultatul dorit, comanda `latex fis` trebuie dată de mai multe ori (de exemplu, crearea unui cuprins necesită două treceri prin fişier, deci va fi nevoie să se mai execute o dată comanda).
3. Înaintea imprimării, este recomandabil să se faca vizualizarea fişierului `.dvi`, folosind comanda:

   ```
   xdvi fis.dvi
   ```

4. După corectarea tuturor posibilelor erori, imprimarea se face astfel:
   - se transformă fişierul `fis.dvi` într-un fişier PostScript cu comanda:

     ```
     dvips -t a4 fis.dvi -o fis.ps
     ```

     care crează fişierul în format PostScript `fis.ps`;
   - se imprimă documentul cu comanda:

     `lpr fis.ps` sau `lp fis.ps`
5. Fişierul în format PostScript poate fi şi el vizualizat, folosind una din comenzile: `ghostview fis.ps` (pe majoritatea sistemelor) sau `xpsview fis.ps` (pe maşina lmnsgi).
   Programul `ghostview` permite şi imprimarea fişierului PostScript şi se recomandă folosirea acestuia, şi nu a comenzii `lpr`, datorită facilităţilor oferite (printre care, vizualizarea şi imprimarea fişierelor comprimate, imprimarea separată a paginilor pare/impare sau a unor pagini selectate de utilizator).

4.2 Structura unui fişier LaTeX

Un fişier LaTeX cuprinde:

- Comentarii: încep cu `%` şi se termină la sfârşitul liniei
- Comenzi LATEX: încep în general cu caracterul `\`
- Conţinutul documentului: text, figuri

Înainte de a scrie orice document, autorul trebuie să îşi clarifice:

- ce tip de document este
  - articol
  - carte
  - raport de cercetare
  - scrisoare, etc.
- titlul, cuprinsul aproximativ
- textul propriu-zis
- aşezarea în pagină – în teorie, ea este stabilită de editorul lucrării (caz în care, pentru a evita erorile şi efortul inutil, se recomandă să se obţină de la acesta un fişier de stil).

LATEX admite comenzi care permit formatarea automată a fiecărei părţi a documentului.

În cele ce urmează sunt prezentate principalele părţi ale unui document şi comenzile LATEX care le "gestionează". Parametrii unei comenzi sunt incluşi între acolade, sau între paranteze drepte în cazul parametrilor opţionali.

Singurele comenzi obligatorii sunt cele referitoare la *tipul de document, începutul* şi *sfârşitul* documentului.

<u>Tipul de document</u> se defineşte cu comanda:

```
\documentstyle[opţiuni]{stil}
```

unde `opţiuni` reprezintă o listă de opţiuni separate prin virgule, care pot fi:

- dimensiune font: `10pt` (implicit), `11pt`, `12pt`
- aliniere ecuaţii (implicit centrate): `fleqn` (stânga)
- aliniere număr ecuaţii (implicit dreapta): `leqno` (stânga)
- includerea instr. de test şi ciclare: `ifthen`
- obtinerea automată a indexului: `makeidx`
- coloane(implicit una): `twocolumn`
- pagini pare/impare diferite: `twoside`
- pagină separată de titlu în stilul `article`: `titlepage`

iar stilurile disponibile sunt:

- `article`
- `book`
- `report`
- `letter`

Părţi document	**Comenzi LaTeX**
Definire tip document	\documentstyle
Definire: titlu	\title
autori	\author
note de subsol	\thanks,
	\footnote
data scrierii doc.	\date
Începutul textului documentului	\begin {document}
Afişarea titlului	\maketitle
Rezumatul lucrării	\abstract
Partea I	\part
Capitolul 1	\chapter
Secţiunea 1.1	\section
Subsecţ. 1.1.1	\subsection
Subsubsecţiune	\subsubsection
....	
Anexe	\appendix
Cuprins	\tableofcontents
Lista de figuri	\listoffigures
Lista de tablele	\listoftables
Sfârşit text document	\end {document}

Definirea paginii de titlu

- Titlu: `\title{text}`

  `text` cuprinde `\\` acolo unde se doreşte trecerea la rând nou

- Autori: `\author{text}`
- Data: `\date{text}`

  Ex: `\date{\today}`

- Trimiteri la note de subsol referitoare la titlu:
  - trimitere la o nouă notă de subsol: `\thanks{text}`

    se recomandă să fie introdusă în text imediat după cuvântul la care se referă

  - trimitere la o notă de subsol existentă: `\footnotemark[n]`,

    cu `n` = numărul notei existente

- Punerea în pagină a titlului: `\maketitle`

  e bine să apară IMEDIAT după `\begin{document}`

- Rezumat:

  `\abstract text abstract \endabstract` sau

`\begin{abstract} text_abstract \end{abstract}`

- Secţiuni ale textului
  - numerotate automat:

    `\part \chapter \section \subsection`
    `\subsubsection`
    toate cu un parametru — textul titlului părţii respective
    Ex: `\part {Partea I - Definiţii generale}`

  - nenumerotate: `\part* \chapter* ....`

  Obs. Numerotarea secţiunilor textului diferă în funcţie de stilul ales.

- Anexe: `\appendix`

  are ca efect schimbarea modului de numerotare automată a capitolelor

- Cuprins, lista de figuri, lista de tabele:

  `\tableofcontents`

  `\listoffigures`

  `\listoftables`

- Referirea la un capitol:

  etichetă: `\label{text_etichetă}`

  referire la o etichetă:

    `\ref{text_etichetă}`
    `\pageref{text_etichetă}` (pagina pe care se găseşte plasat obiectul referit prin etichetă)

La crearea titlului, părţilor, listelor, cuprinsului, etc., LATEX foloseşte texte predefinite în limba engleză. Acestea sunt:

Comanda LATEX	**Semnificaţie**	**Valoare implicită**	**Stilurile în care este disponibilă**
`\partname`	nume parte	Part	book, report
`\chaptername`	nume capitol	Chapter	book, report
`\refname`	nume bibliografie	References	article
`\bibname`	nume bibliografie	Bibliography	book, report
`\appendixname`	nume anexe	Appendix	toate
`\abstractname`	nume rezumat	Abstract	toate
`\indexname`	nume index	Index	toate
`\contentsname`	nume cuprins	Contents	toate
`\figurename`	nume figură	Figure	toate
`\listfigurename`	nume listă figuri	List of Figures	toate
`\tablename`	nume tabel	Table	toate
`\listtablename`	nume listă tabele	List of Tables	toate

Schimbarea lor se poate face cu comanda:

```
\renewcommand <par1> <par2>
```

unde:

`par1` = numele comenzii
`par2` = noua definiţie a comenzii

Ex: `\renewcommand {\chaptername}{Capitolul}`

4.3 Comenzi LaTeX

În general sunt reprezentate de: caracterul backslash (\) urmat de un cuvânt (o singură cifră, o literă sau mai multe). Există două tipuri de comenzi:

- Cu efect localizat - execută o acţiune, apoi se încheie, cum ar fi de exemplu comenzile de tipărire a unor secvenţe de caractere;
  Exemple:

  `\%` - tipăreşte %, încheie comanda

  `\today` - tipăreşte data de azi, încheie comanda

  `\LaTeX` - tipăreşte LaTeX, iese

  `\par` - trece la paragraf nou, iese

- Cu efect global - efectul lor se încheie doar când se dă o altă comandă cu efect "contrar";
  Ex: comenzile de schimbare a fontului:

  `\tt` - schimbă fontul în "typewriter"

  `\bf` - schimbă fontul în "bold"

Există şi comenzi care aşteaptă o listă de parametri. Parametrii pot fi:

- un singur caracter care urmează imediat după numele comenzii, ca în exemplele:

  `\a` - comanda "tipăreşte accent" (ea aşteaptă un parametru)

  `\a=` - tipăreşte accentul notat =

- un şir de caractere între acolade

  Ex: `\hspace {1cm}` - lasă un spaţiu orizontal de 1 cm

Pentru a localiza efectul unei comenzi cu efect global, se foloseşte gruparea - includerea între acolade a porţiunii pentru care se doreşte modificarea prin comandă.

Ex: `Font normal {\bf font bold} font normal.`

Atenţie! Tot acolade se folosesc, după cum s-a văzut mai sus, şi pentru marcarea parametrilor! Parametrii, deşi sunt puşi între acolade, nu reprezintă grupuri, deci, dacă în interiorul unui parametru apare o comandă cu efect global, efectul ei va continua şi după acolada închisă.

4.4 Moduri de lucru LATEX

În orice punct al documentului LATEX, ne putem afla în unul din trei moduri:

- modul paragraf - acesta este cel "implicit", în el se poate scrie textul;
- modul matematic - permite scrierea de formule matematice;
- modul LR - analog cu paragraf, dar în acest mod LATEX nu verifică încadrarea în spaţiul disponibil, cum face în mod paragraf.

În fiecare mod de lucru se pot da comenzi specifice.

4.4.1 Modul paragraf

Este modul de lucru implicit.

Comenzi - grupate după efectul lor:

- setare font:

  `\rm` - Roman

  `\it` - *Italic*

  `\bf` - **Bold**

  `\sf` - Sans Serif

  `\sc` - SMALL CAPITALS

  `\sl` - *Caractere Inclinate (slanted)*

  `\tt` - `Typewriter`

  `\em` - trecerea de la roman la *italic şi* invers

- setare dimensiune font:

  `\tiny, \scriptsize, \footnotesize, \small,`
  `\normalsize, \large, \Large, \LARGE, \huge, \Huge.`

  cu efectul:

  Font, Font, Font, Font, Font, Font, Font, Font, Font, Font.

  Dimensiunea propriu-zisă a acestor fonturi depinde de opţiunea (`10pt, 11pt, 12pt`) din `\documentstyle`.

- printare caractere speciale (vezi paragraful 4.7.1 şi manualul de utilizare pentru lista completă)
- printare linii şi spaţii de diverse dimensiuni:

  `-` - linie de despărţire în silabe

  `--` – linie din matematică

  `---` — linie (punctuaţie)

`\,` - spaţiu mic, utilizat de exemplu între ghilimelele simple şi cele duble

`\␣` - spaţiu (caracterul backslash urmat de un spaţiu)

`\/` - spaţiu pentru a nu permite ligatura unor cuvinte

`\@` - spaţiu de sfârşit de propoziţie, dacă următoarea propoziţie începe cu literă mică

`\frenchspacing` - suprimă spaţiul suplimentar după punctuaţie

`\nofrenchspacing` - elimină efectul comenzii `\frenchspacing`

`\noindent` - suprimă alineatul de început de paragraf

`\par` - linie liberă

`\hspace` - spaţiu orizontal, uneori nu are efect la început şi sfârşit de paragraf

`\hspace*` - ca mai sus, apare oriunde

`\vspace` - spaţiu vertical

`\vspace*` - forma robustă a celei de mai sus

`\hfill` - spaţiu orizontal de lungime variabilă (cât să completeze linia);

`\vfill` - spaţiu vertical de dimensiune variabilă (cât să completeze pagina);

`\dotfill` - ca şi `\hfill` dar produce puncte

`\hrulefill` - ca şi `\hfill` dar produce o linie orizontală

`\smallskip, \medskip, \bigskip` - spaţii verticale (depind de stilul documentului)

`\addvspace` - adaugă un spaţiu vertical

- printare note marginale:

  `\marginpar`

- printare dreptunghiuri:

  `\rule{dimx}{dimy}` sau

  `\rule[dy]{dimx}{dimy}` (supraînălţat faţă de baza textului cu dy)

- setare opţiuni pentru rânduri:

  `\\` - rând nou, restul liniei rămâne gol

  `\newline` - echivalent cu `\\`

  `\linebreak` - linie nouă, caracterele rămase se spaţiază ca să ocupe întreaga lăţime

  ~ - interzice ruperea liniei în acel loc

  `\nolinebreak` - interzice ruperea liniei în acel loc

- setare opţiuni pentru pagini:

  `\flushbottom` - adăugare de spaţiu pe verticală a.î. toate paginile să aibă aceeaşi lungime

`\raggedbottom` - efect invers

`\newpage` - trecerea la pagină nouă

`\pagebreak` - trecerea la pagină nouă

`\nopagebreak` - interzice trecerea la pagină nouă

`\samepage` - păstrarea unui set de linii pe aceeaşi pagină

`\clearpage` - trecerea la pagină nouă, cu punerea tabelelor şi figurilor rămase pe o pagină fără text

4.4.2 Modul matematic

Intrare în mod matematic:

- formulă în text:

  `$ formulă $`

  sau

  `\begin{math} formulă \end{math}`

- formulă nenumerotată, în paragraf separat:

  `$$ formulă $$`

  sau

  `\begin{displaymath} formulă \end{displaymath}`

  sau

  `\[ formulă \]`

- formulă numerotată în paragraf separat – vezi paragraful 4.5 → ecuaţii.

Comenzi disponibile:

- indici: `_` `^` - indice inferior, superior
- fracţie: `\frac{numărător} {numitor}`
- radical: `\sqrt{x}, \sqrt[n]{x}`
- funcţii matematice: `\sin, \cos, \arccos, \lim, \min`, etc.
- modulo: `\bmod, \pmod`
- semne speciale:

  `\sum` - sigma

  `\int` - integrală

  `\lim` - limită

  `\cdots, \vdots, \ddots` - puncte centrate, verticale sau diagonale

  `\ldots` - trei puncte la baza liniei

- litere greceşti: `\alpha, \beta`, .. $\alpha, \beta, \ldots$

- setare font :

  `\mathcal` - $\mathcal{CALIGRAFIC}$, disponibil doar pentru literele mari

  `\mathit` - *mathitalic* (implicit)

  `\mathbf` - bold **bold**

  `\mathrm` - Roman, drept

- delimitatori:

  `\left( .... \right)`

  `\left{ .... \right}` sau

  `\left{ ... \right.` pentru a nu mai afişa şi paranteza din dreapta

- linii, acolade:

  `\overline{text}` - linie deasupra: $\overline{text}$

  `\underline` - subliniere: $\underline{text}$

  `\overbrace` - acoladă deasupra: $\overbrace{text}$

  `\underbrace` - acoladă dedesubt: $\underbrace{text}$

  `\vec` - simbol de vector (sageată orizontală) deasupra: $\vec{v}$

  `\imath` - i fără punct: $\imath$

  `\jmath` - j fără punct: $\jmath$

  `\hat, \widehat` - căciuliţă deasupra: $\hat{v}, \widehat{v}$

  `\tilde, \widetilde` - tildă deasupra: $\tilde{v}, \widetilde{v}$

  (vezi pag. 78-79 din [71]).

- spaţiere:

  `\;` - spaţiu mare

  `\!` - spaţiunegativ ingust

  `\quad` - spaţiu egal cu de 6 ori `\,` (vezi paragraful 4.4.1)

  `\qquad` - spaţiu egal cu de 10 ori `\,` (vezi paragraful 4.4.1)

4.4.3 Modul LR

În acest mod, orice "obiect" LaTeX (de exemplu: o literă, un fragment de text, o ecuaţie, etc.) sunt tratate ca nişte "cutii" despre care LaTeX cunoaşte doar dimensiunile. Avantaj: comenzile care culeg text în acest mod pot apărea la rândul lor în orice mod (paragraf, matematic sau LR).

În acest mod se intră prin comenzi caracteristice:

- Culegere text într-o "cutie":

\mbox

\makebox

\fbox

\framebox

\raisebox

- Parametri ai stilului:

\fboxrule

\fboxsep

- Salvare cutie / citire "cutie" salvată:

\newsavebox

\sbox

\usebox

4.5 Cadru LATEX

Delimitează o entitate (ecuaţie, tabel, figură, fragment de text, etc.) şi reprezintă o parte a documentului care este aşezată în pagină într-un mod special, caracteristic entităţii respective.

Formatul general al comenzii `environment` (în traducere *cadru* este:

```
\begin {nume_environment} text \begin {nume_environment}
```

Aceste structuri pot fi imbricate, având grijă ca ultimul deschis să fie închis primul.

Cadruri disponibile (puse în loc de `nume_environment`):

- document: `document`
- rezumat: `abstract`
- citate:

quote

quotation

verse

- alinieri:

flushright

center

- păstrarea nemodificată a textului: `verbatim`
- liste:

itemize

`enumerate`

`description`

`list`

- tabele:

  `tabbing`

  `tabular`

  `table` (env. mobil)

- ecuaţii:

  `displaymath`

  `equation`

  `array`

  `eqnarray`

- desene:

  `picture`

  `figure` (env.mobil)

- "pagini" miniatură: `minitext`
- bibliografie: `thebibliography`
- index: `theindex`

4.6 Macrodefiniţii LaTeX

Utilizatorul poate extinde şi configura LaTeX, prin definirea de noi comenzi şi cadruri în cadrul documentului şi prin modificare celor existente.

- Comenzi noi

  fără parametri: `\newcommand{\comandă_nouă} {definiţie}`

  unde `comanda_nouă` este: o cifră SAU mai multe litere

  cu parametri: `\newcommand {\comandă_nouă}[n]{definiţie}`

  unde `n` este numărul de parametri, mai mic decât 9. Parametrul numărul k este utilizat în definiţie sub forma: `#k`

- Redefinire comenzi:

  `\renewcommand{\comandă_veche}[n]{definiţie_nouă}`

- Environments noi

  `\newenvironment ...`

- Facilitate scriere teoreme:

  `\newtheorem ...`

4.7 Aspecte speciale ale utilizării LaTeX

4.7.1 Caractere speciale

Următoarele caractere au semnificaţii speciale în cadrul comenzilor LaTeX şi nu vor apărea ca atare în text:

`$ & # _ { } ~ ^ " \ |`

iar la scrierea cu anumite fonturi şi <>.

Uneori, nici blanc sau <CR> nu apar cum ne-am aştepta:

- mai mult decât un blanc = un blanc
- un singur <CR> = blanc
- mai mult de 2<CR> = un <CR>

Cum se pot obţine aceste caractere: unele dintre ele, precedându-le de un backslash `\`

- diverse `\$ \& \% \# \_ \{ \} \blanc`
- linie nouă: 2 `<CR>`
- backslash:

  `$\backslash$` (în mod matematic, vezi moduri LaTeX mai sus)

  `\tt\char 39` (în mod paragraf)

4.7.2 Accente

Se folosesc pentru obţinerea literelor româneşti (sau din alte limbi). Pentru limba română:

`\^A, \^a` : Â â
`\uA, \ua` : Ă ă
`\cS, \cs`: Ş ş
`\cT, \ct`: Ţ ţ

Caz special: `\^i` ar da un rezultat nedorit: î (apare şi punctul şi căciuliţa). Accentul ar trebui aplicat literei i fără punct, definit ca `\i`. În consecinţă, `\^\i` va avea efectul dorit: î.

În reţeaua LMN este instalat stilul LaTeX `rom.sty` (realizat de I. Munteanu prin adaptarea modelului propus în [71]), stil care permite introducerea diacriticelor româneşti într-un mod mult mai simplu.

Pentru a folosi acest stil, în lista de opţiuni a comenzii `\documentstyle` trebuie introduse şi opţiunile

```
[romanian,rom]
```

(în această ordine). Folosind stilul `rom`, diacriticele româneşti pot fi dactilografiate astfel:

´a – ă
´A – Ă
´i – î
´I – Î
´s – ş
´S – Ş
´t – ţ
´T – Ţ
´w – â

Se observă că ′ (apostroful) devine în acest stil o *comandă* şi nu un caracter obişnuit.

În acelaşi fişier de stil sunt definite două comenzi, care permit definirea caracterului ´ ca fiind o comandă pentru introducerea diacriticelor, respectiv un caracter obişnuit:

\rom – caracterul ′ devine o comandă
\norom – caracterul ′ devine un caracter afişabil.

Se recomandă, de exemplu, introducerea comenzii \norom înainte de comanda de generare a bibliografiei (în fişierele de bibliografie apostroful trebuie să fie un caracter afişabil), şi a comenzii \rom imediat după comanda de generare a bibliografiei (deoarece, în documentele româneşti, şi anexele conţin diacritice).

4.7.3 Despărţirea în silabe

Regulile de despărţire în silabe depind de limba utilizată în text. Aceasta se poate redefini cu comanda:

\language<număr_de_ordine>

Ex: \language0 = limba engleză

Pentru a despărţi în silabe un cuvânt (efect local) se foloseşte comanda \-

ex: des\-partire

Acest lucru ar putea fi necesar dacă în urma procesării documentului cu LaTeX apare eroarea:

```
Overfull xxxpt too wide
```

adică, există un cuvânt prea lung, care nu poate fi trecut integral pe linia următoare fără a spaţia mult prea mult rândul curent.

4.8 Structura unei pagini

Pentru definirea lăţimilor, lungimilor şi distanţelor, LaTeX foloseşte parametri de stil. Ei au valori implicite, dar pot fi modificaţi de către utilizator. Pentru o listă completă puteţi consulta [71], pag. 89.

Modificarea lor se face cu comanda:

```
\setlength{\par_de_stil}{dimens}
```

unde `par_de_stil` poate fi numele unei dimensiuni: lungimea textului pe pagină (`\textheight`), lăţimea textului pe pagină (`\textwidth`), distanţa între paragrafe (`\parskip`), margini (dreapta, stânga, sus, jos), dimensiune header sau footer, etc.

4.9 Fişiere rezultate în urma procesării cu LaTeX

`.tex`	-	fişier "sursă"
`.dvi`	-	fişier de ieşire
`.aux`	-	fişier cu informaţii auxiliare (folosit la a doua procesare pentru a rezolva referinţe încrucişate
`.toc`	-	fişier cu informaţii necesare alcătuirii cuprinsului
`.log`	-	fişier conţinând raportul asupra procesării fişierului `.tex`, erori, folosirea resurselor de calcul
`.lof`	-	fişier asemănător cu `.toc`, dar pentru lista figurilor
`.lot`	-	fişier asemănător cu `.toc`, dar pentru lista tabelelor
`.idx`	-	fişier cu informaţii necesare alcătuirii indexului
`.glo`	-	fişier cu informaţii necesare alcătuirii glosarului de termeni

4.10 Câteva exemple

4.10.1 Fişierul `latex_head.tex`

Mai jos este prezentată structura unui fişier header, care conţine comenzi de punere în pagină şi care poate fi adaptat şi inclus în orice document LaTeX.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FISIER latex_head.tex
% Contine comenzi de punere in pagina
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% -- stilul paginii --
\pagestyle{plain}

% -- spatierea paragrafelor --
```

```
\setlength{\parskip}{1.5ex plus0.5ex minus0.5ex}
\setlength{\parindent}{1em}
\renewcommand{\baselinestretch}{1}

% -- formatul paginii --
\raggedbottom
% flushbottom              % varianta la \raggedbottom

\setlength{\textwidth}{16cm}
\setlength{\textheight}{25cm}
\setlength{\topmargin}{0cm}
\setlength{\oddsidemargin}{1cm}
\setlength{\evensidemargin}{0cm}
\setlength{\headheight}{0cm}
\setlength{\headsep}{0cm}

% -- figuri --
\newcounter{myfig}              % contor pentru figuri

\newcommand{\mycaption}[2] {    % comanda pentru numerotarea figurilor
        \refstepcounter{myfig}  % creste contorul
        \label{#2}              % #2 eticheta pentru referiri ulterioare
        \begin{center}
        {\bf Fig. \themyfig. }{#1}    % #1 titlul figurii
        \end{center}
}

\newcommand{\PSFIG}[5]          % figura ps numerotata,
{                               % plasata in pagina de LaTeX
        \begin{figure}{#5}      % #5 este una din literele h, t, b
        \centerline{\psfig{figure=#1,height=#2}}
                                % #1 este numele fisierului ps
                                % #2 -dimensiune
        \caption{#3}{#4}        % #3 - titlul #4 - eticheta
}

\newcommand{\PSFIGH}[4]          % figura ps numerotata,
{                                % plasata in pagina in punctul apelului
        \vspace{0.5cm}
        \par
        \centerline{\psfig{figure=#1,height=#2}}
        \par                    % #1 nume fisier ps;#2 -dimensiune
        \samepage
        \mycaption{#3}{#4}      % #3 - titlul #4 - eticheta
}
```

4.10.2 Un exemplu de document

Comenzi de formatare LaTeX

– Exemple de utilizare –

Nume Autor

28 iulie 1998

Cuprins

1 Intrare în mod matematic **1**

2 Formule matematice **2**

3 Tabele **3**

1 Intrare în mod matematic

Formulele matematice pot fi scrise în cadrul textului sau pe linie separată.

1.1 Formule în text

Formulele se pot scrie în cadrul textului $a_1 = x^2$, anumite fonturi fiind în acest caz mai mici (selectate automat de LaTeX).

1.2 Formule pe linie separată

Alternativ, formulele se pot scrie pe linie separată:

$$a_1 = x^2.$$

Acelaşi rezultat se obţine şi astfel:

$$a_1 = x^2.$$

De multe ori e nevoie de ecuaţii numerotate:

$$a_1 = x^2. \tag{1}$$

2 Formule matematice

2.1 Integrale

$$\mathcal{I} = \int_a^{a+b} \left(x^2 + \sin(x) + \mathrm{e}^x \right) \mathrm{d}x.$$

2.2 Vectori

$$\mathrm{rot}\ \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}, \qquad < \mathbf{a}, \mathbf{b} >= \mathbf{a} \cdot \mathbf{b}.$$

2.3 Sumă, produs

$$S = \sum_{k=1}^{\infty} a_k^2 = \sum_{k=1}^{\infty} \left(\frac{1}{2k+1} \right)^2, \qquad P = \prod_{k=1}^{m} S_k.$$

2.4 Utilizarea parantezelor

Comparaţi următoarele două formule:

$$y = (x(2x^2 + x(a+c) + b^2) + \frac{b^2}{c^2})$$

$$y = \left(x \left(2x^2 + x\left(a+c\right) + b^2 \right) + \frac{b^2}{c^2} \right)$$

2.5 Funcţii matematice

$$f(x) = \begin{cases} x, & x < 0 \\ x^2, & x \geq 0 \end{cases}$$

$$E = \sin(x) + \tan(x) + \ln(x).$$

$$\lim_{x \to \infty} \frac{x+1}{2x+7} = \frac{1}{2}, \quad x \in \mathbb{R}.$$

2.6 Matrice

$$A = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

Centrat	Centrat	Centrat
Centrat	Centrat	Centrat

Tabelul 1: O explicaţie pentru tabel

2.7 Sisteme de ecuaţii

$$\begin{aligned} x^2 + 2x + y &= 0; & (2) \\ 2y^2 + 2y + 3x + 7xy &= 1. & (3) \end{aligned}$$

sau pentru a numerota o singură dată:

$$\begin{aligned} x^2 + 2x + y &= 0; \\ 2y^2 + 2y + 3x + 7xy &= 1. \end{aligned} \quad (4)$$

$$\begin{array}{l} x^2 + 2x + y = 0; \\ 2y^2 + 2y + 3x + 7xy = 1. \end{array} \quad (5)$$

Pentru a alinia ecuaţiile la alt caracter decât semnul "=" se procedează ca în ecuaţiile (6):

$$\begin{array}{rcl} x^2+ & 2x + y & = 0; \\ 2y^2+ & 2y + 3x + 7xy & = 1 \end{array} \quad (6)$$

2.8 Diverse

Nu ştim cum să facem litere greceşti bold: φ φ.

Observaţi diferenţa dintre: aa...a şi $aa\cdots a$.

$$||\mathbf{v}||_2 \stackrel{def}{=} \sqrt{\sum_k v_k^2}$$

3 Tabele

Un tabel plasat în punctul unde este apelat:

Aliniat stânga	Centrat	Lăţime 5cm	Aliniat dreapta
Bucureşti	S	Aici putem scrie un text mai lung care va fi aliniat la ambele margini	1.000
Timişoara	V	Text	2.000
Cluj	E	Text	3.000

Se pot defini şi tabele numerotate (tabelul 1), plasate automat de LaTeX.

Şi iată cum a fost realizat:

```
\documentstyle[psfig,twoside,12pt,a4,romanian,rom]{article}

\input{latex_head.tex}

% Definirea unor noi comenzi; pot fi puse in latex_head.tex
\newcommand{\mr}[1]{{\mathrm{#1}}}
\newcommand{\mb}[1]{{\mathbf{#1}}}
\newcommand{\mc}[1]{{\mathcal{#1}}}

\title{{\huge \bf Comenzi de formatare \LaTeX}\ \\ %\newline
          {\Large\sl -- Exemple de utilizare --}}
\author{Nume Autor}
\date{\today}

%=====================================================================
%=====================================================================

\begin{document}        % inceputul documentului
%\large
\maketitle              % tipareste titlul, autorul ´si data

\setcounter{tocdepth}{1}
\tableofcontents        % tipareste cuprinsul lucrarii

\section{Intrare ´in mod matematic}   % inceputul unei sectiuni
                                      % sectiunile se numeroteaza automat

Formulele matematice pot fi scrise ´in cadrul textului
sau pe linie separat´a.

\subsection{Formule ´in text}         % inceputul unei subsectiuni
Formulele se pot scrie ´in cadrul textului $a_1 = x^2$, anumite
fonturi fiind ´in acest caz mai mici (selectate automat de \LaTeX).

\subsection{Formule pe linie separat´a} % o noua subsectiune
Alternativ, formulele se pot scrie pe linie separat´a: $$a_1 = x^2.$$

Acela´si rezultat se ob´tine ´si astfel:
\begin{displaymath}
a_1 = x^2.
\end{displaymath}

De multe ori e nevoie de ecua´tii numerotate:
\begin{equation}
a_1 = x^2.
\end{equation}
```

```
\section{Formule matematice}

\subsection{Integrale}
$$
{\mathcal{I}} = \int_a^{a+b} \left(x^2 + \sin(x) + \mr e^x \right)
\mr dx.
$$

\subsection{Vectori}
$$ \mr{rot} \ \mb E = - \frac{\partial \mb B}{\partial t},
\qquad \qquad
< \mb a, \mb{b} > = \mb a \cdot \mb b.
$$

\subsection{Sum´a, produs}
$$ S = \sum_{k=1}^{\infty} a_k^2 =
\sum_{k=1}^{\infty} \left( \frac{1}{2 k +1} \right)^2,
\qquad
P = \prod_{k=1}^{m} S_k.
$$

\subsection{Utilizarea parantezelor}
Compara´ti urm´atoarele dou´a formule:
$$
y = ( x (2 x^2 + x (a + c) + b^2) + \frac{b^2}{c^2} )
$$

$$
y = \left( x \left(2 x^2 + x \left(a + c \right) + b^2 \right) +
\frac{b^2}{c^2} \right)
$$
\subsection{Func´tii matematice}
$$
f(x) = \left\{
\begin{array}{ll}
x,   & x < 0 \\
x^2, & x \ge 0
\end{array}
\right.
$$

$$
E = \sin(x) + \tan(x) + \ln(x).
$$

$$
\lim_{x \to \infty} \frac{x+1}{2 x+7} = \frac{1}{2}, \ \ x \in \mr{I\!R}.
$$
```

```
\subsection{Matrice}
$$
A = \left[
\begin{array}{cccc}
1 & 0 & \cdots & 0 \\
0 & 1 & \cdots & 0 \\
\vdots & & \ddots & \vdots \\
0 & 0 & \cdots & 1
\end{array}
\right]
$$

\subsection{Sisteme de ecua´tii}
\begin{eqnarray}
x^2 + 2x + y = 0; \\
2 y^2 + 2y + 3x + 7xy  =  1.
\end{eqnarray}
sau pentru a numerota o singur´a dat´a:
\begin{eqnarray}
x^2 + 2x + y = 0; \\ \nonumber
2 y^2 + 2y + 3x + 7xy  =  1.
\end{eqnarray}
%  astfel reducem spatiul liber intre doua ecuatii
\begin{equation}
\begin{array}{l}
x^2 + 2x + y = 0; \\
2 y^2 + 2y + 3x + 7xy  =  1.
\end{array}
\end{equation}

Pentru a alinia ecua´tiile la alt caracter dec\^at semnul ``="
se procedeaz´a ca ´in ecua´tiile (\ref{ec1}):
\begin{eqnarray}
\label{ec1}
x^2 + & 2x + y & =  0 ; \\ \nonumber
2 y^2 + & 2y + 3x + 7xy & =  1
\end{eqnarray}

\subsection{Diverse}
Nu ´stim cum s´a facem litere grece´sti bold:
$
\mb{\varphi}  \ \ \varphi
$.

Observa´ti diferen´ta dintre: aa$\ldots$a ´si aa$\cdots$a.
```

```
$$
|| \mb v ||_2 \stackrel{def}{=} \sqrt{\sum_k v_k^2}
$$

\section{Tabele}

Un tabel plasat ´in punctul unde este apelat:

\begin{tabular}{|l|c||p{5cm}||r|}
\hline
Aliniat st\^anga & Centrat & L´a´time 5cm & Aliniat dreapta \\
\hline
Bucure´sti & S & Aici putem scrie un text mai lung
              care va fi aliniat la ambele margini & 1.000 \\ \hline
Timi´soara & V & Text & 2.000 \\ \hline
Cluj       & E & Text & 3.000 \\
\hline
\end{tabular}

Se pot defini ´si tabele numerotate (tabelul \ref{tabelcentrat}), plasate
automat de \LaTeX.

\begin{table}[t]
\begin{center}
\begin{tabular}{|ccc|}
\hline
Centrat & Centrat & Centrat \\ \hline
Centrat & Centrat & Centrat \\ \hline
\end{tabular}
\end{center}
\caption{O explica´tie pentru tabel}
\label{tabelcentrat}
\end{table}
\end{document}
```

4.10.3 Un exemplu de articol în stilul IEEE Trans. on Magnetics

Parameter Extraction for Microwave Devices Based on 4SID Techniques

Irina Munteanu and Daniel Ioan, members, IEEE
"POLITEHNICA" University of Bucharest, Numerical Methods Laboratory – Electrical Engineering Dept.
Spl. Independentei 313, 77206 Bucharest, Romania

***Abstract*—Microwave devices parameter extraction is achieved by a field analysis of the device, followed by a procedure capable to extract lumped circuit parameters from the time domain information. The proposed algorithm is based on the 4SID (Subspace–based State–Space System IDentification) techniques, followed by an optimal order reduction.**

***Index terms*—Microwave, parameter extraction, Hankel norm, 4SID technique.**

I. Introduction

In order to reduce the simulation time, microwave devices are often represented by approximate lumped circuits. Subsequently, SPICE–like simulators can be used to analyze complex configurations.

For circuit parameter extraction, several techniques are currently used, among them the Krylov Subspace [1], Padé via Lanczos, and asymptotic waveform evaluation [2] techniques.

II. Problem Formulation and Solution

Under certain boundary conditions [3], a microwave device can be treated as an Electromagnetic Circuit Element (**ECE**).

In its domain $\mathcal{D}$, linear media are considered:

$$\nabla \times \mathbf{H} = \sigma \mathbf{E} + \varepsilon \frac{\partial \mathbf{E}}{\partial t}, \qquad \nu \nabla \times \mathbf{E} = -\frac{\partial \mathbf{H}}{\partial t}. \qquad (1)$$

For each of the n terminals placed on the boundary $\partial\mathcal{D}$, a voltage and a current can be properly defined. As proven in the full paper, the solution of (1) with ECE boundary conditions can be expressed as a series of generalized eigenfunctions of the $\nabla \times (\nu \nabla \times \cdot)$ operator, which corresponds to an infinite RLC circuit.

What we look for, is an n–pole lumped circuit which is the best finite approximation for the device. It can be obtained by the following **algorithm**:

Step 1: The problem (1) together with the ECE boundary conditions is solved only once by a time–domain numerical method, at arbitrary excitation (the one used in device's real functioning is recommended);

Step 2: The 4SID algorithm [4] is used to generate a system S, equivalent to the device in what concerns the terminal behaviour. Using the best Hankel–norm approximation a reduced–order system S' is generated [3];

Step 3: The transfer model of the system S', together with Foster synthesis provide the lumped circuit.

The parameter extraction algorithm was first applied to a problem with known analytical solution,

A bended strip, placed on a substrate and enclosed in a metallic box has also been analyzed. Numerical results will be presented in the full paper.

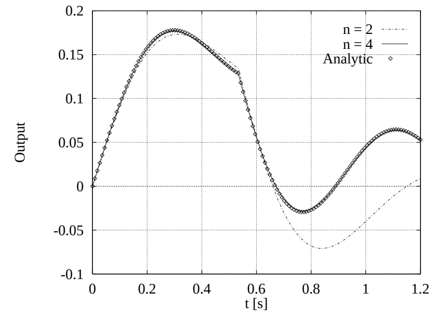


Fig. 1. Output of the initial and reduced systems

III. Conclusions

The proposed technique appears to be an efficient alternative for determining lumped circuit parameters for a class of general distributed–parameters devices. It can be used for extracting parameters valid over a wide range of frequencies with only one time–domain field simulation. Results obtained so far are very promising.

References

[1] M. Celik, A. C. Cangellaris, and A. Yaghmour, "An all–purpose transmission–line model for interconnect simulations in SPICE," *IEEE Trans. Microwave Theory Tech.*, vol. 45, pp. 1857–1867, Oct. 1997.

[2] J. E. Bracken and Z. J. Cendes, "Evaluation for S-domain solution of electromagnetic waves," in *Proc. 11th COMPUMAG Conf.*, (Rio de Janeiro, Brazil), pp. 207–208, Nov. 1997.

[3] D. Ioan, I. Munteanu, and C.-G. Constantin, "The best approximation of the field effects in electric circuits coupled problems," in *Proc. 11th COMPUMAG Conf.*, (Rio de Janeiro, Brazil), pp. 747–748, Nov. 1997.

[4] Y. Cho, G. Xu, and T. Kailath, "Fast identification of state-space models via exploitation of displacement structure," *IEEE Trans. AC-26*, no. 10, pp. 2004–2017, 1994.

Şi iată cum a fost realizat:

```
% Rezumat ´in stilul IEEE Transactions on Magnetics
\documentstyle[IEEEmagn,psfig]{article}

% Definitii
\newcommand{\half}%
{\raisebox{2.5pt}{\scriptsize 1}{\small /}\raisebox{-1pt}{\scriptsize 2}}

\newcommand{\B}{{\bf B}}
\newcommand{\E}{{\bf E}}
\newcommand{\U}{\overline{u}}
\newcommand{\Fi}{\overline{\varphi}}
\newcommand{\Ep}{\overline{\varepsilon}}
\newcommand{\M}{\overline{\mu}}

\newcommand{\Be}{{\mathbf B}}
\newcommand{\Ee}{{\mathbf E}}
\newcommand{\He}{{\mathbf H}}
\newcommand{\De}{{\mathbf D}}
\newcommand{\Je}{{\mathbf J}}
\newcommand{\Do}{${\mathcal D}$}

% -- keywords
\def\keywords#1{\vspace*{3.5ex}{\small \boldmath \indent
            $Index\ terms$\/}---\ignorespaces {\small \bf #1}}

% -- figures --
\newcommand{\PSFIG}[4]
{
\begin{figure}
        \centerline{\psfig{figure=#1,height=#2}}
        \caption{#3}
        \label{#4}
\end{figure}
}


\begin{document}

\boldmath \title{Parameter Extraction for Microwave Devices Based on
4SID Techniques} \unboldmath

\author{Irina Munteanu and Daniel Ioan, members, IEEE\\
{\small "POLITEHNICA" University of Bucharest, Numerical Methods Laboratory --
  Electrical Engineering Dept. }\\
  {\small Spl. Independentei 313, 77206 Bucharest, Romania} }

%% In articolul final apar si aceste linii
```

```
%%
%% \thanks{%
%% \newline \indent I.\ Munteanu, e-mail irina@lmn.pub.ro.
%% \newline \indent D.\ Ioan, e-mail daniel@lmn.pub.ro.
%% \newline \indent This work was partially supported by the Romanian Ministry
%% of Education under the CNCSU 106 grant.
%% }

\maketitle

\abstract{%
Microwave devices parameter extraction is achieved by a field analysis of the
device, followed by a procedure capable to extract lumped circuit parameters
from the time domain information.  The proposed algorithm is based on the
4SID (Subspace--based State--Space System IDentification) techniques,
followed by an optimal order reduction.
}

\keywords{Microwave, parameter extraction, Hankel norm, 4SID technique.}

\section{Introduction}
In order to reduce the simulation time, microwave devices are often
represented by approximate lumped circuits. Subsequently, SPICE--like
simulators can be used to analyze complex configurations.


For circuit parameter extraction, several techniques are currently used,
among them the Krylov Subspace \cite{celik97a}, Pad\'{e} via
Lanczos,  and asymptotic waveform evaluation \cite{bracken97a} techniques.

\section{Problem Formulation and Solution}

Under certain boundary conditions \cite{ioan97n}, a microwave device can be
treated as an {Electromagnetic Circuit Element} ({\bf ECE}).

In its domain \Do, linear media are considered:
%
\begin{equation}
\begin{array}{l}
\nabla\times\He = \sigma \Ee + \varepsilon\frac{\partial\Ee}{\partial t},\qquad
\nu\nabla\times\Ee = -\frac{\partial\He}{\partial t}.
\end{array} \label{ec_gen}
\end{equation}
%
For each of the $n$ terminals placed on the boundary $\partial$\Do, a voltage
and a current can be properly defined.  As proven in the full paper, the
solution of (\ref{ec_gen}) with ECE boundary conditions can be expressed as a
series of generalized eigenfunctions of the $\nabla \times (\nu \nabla \times
```

```
\mathbf\cdot)$ operator, which corresponds to an infinite RLC circuit.

What we look for, is an $n$--pole lumped circuit which is the best finite
approximation for the device. It can be obtained by the following {\bf
algorithm}:

{\bf Step 1:} \ \ \  The problem (\ref{ec_gen}) together with the ECE boundary
conditions is solved only once by a time--domain numerical method, at
arbitrary excitation (the one used in device´s real functioning is
recommended);

{\bf Step 2:} \ \ \  The 4SID algorithm \cite{Cho} is used to generate a
system $S$, equivalent to the device in what concerns the terminal behaviour.
Using the best Hankel--norm approximation  a reduced--order system $S´$ is
generated  \cite{ioan97n};

{\bf Step 3:} \ \ \  The transfer model of the system $S´$, together with
Foster synthesis provide the lumped circuit.

The parameter extraction algorithm was first applied to a problem with known
analytical solution.

\PSFIG{sol.eps}{4.7cm}{Output of the initial and reduced systems}{sol}

A bended strip, placed on a substrate and enclosed in a metallic box has also
been analyzed.  Numerical results will be presented in the full paper.

\section{Conclusions}
The proposed technique appears to be an efficient alternative for determining
lumped circuit parameters for a class of general distributed--parameters
devices.
It can be used for extracting parameters valid over a wide range of
frequencies with only one time--domain field simulation. Results obtained so
far are very promising.
\bibliographystyle{ieeetr}
\bibliography{rio97,ref}

\end{document}
```

4.11 Gestionarea referinţelor bibliografice folosind BIBTEX

Referinţele bibliografice se pot introduce direct, în interiorul fişierului `tex`. În acest fel însă, apare dezavantajul că o aceeaşi referinţă trebuie introdusă de mai multe ori, în fiecare fişier `.tex` în care este folosită. Cum fiecare editură are propriul său format al citărilor, aceeaşi informaţie (referinţa bibliografică) trebuie pusă în pagină în moduri diferite.

De aceea, se recomandă utilizare unor fişiere BIBTEX, care sunt de fapt baze de date cu referinţe bibliografice şi de unde LATEX îşi poate prelua numai acele referinţe care sunt citate. În plus, o aceeaşi referinţă poate fi pusă în pagină în mod diferit, în diferite fişiere LATEX.

Un alt avantaj al utilizării BIBTEX constă în aceea că editarea bazei de date (adăugare, ştergere, modificare a unei referinţe) se poate face eficient cu utilitarele **BibCard** şi **BibView**, care au o interfaţă grafică prietenoasă cu utilizatorul (figura 4.1).

Fig. 4.1: Interfaţa grafică a programului **BibCard**

4.11.1 Structura unui fişier BIBTEX

Fişierele BIBTEX sunt fişiere text care conţin *înregistrări*, câte una pentru fiecare referinţă. Un astfel de fişier are extensia `.bib`.

O **înregistrare** BIBTEX are structura:

`@`*tip* {*cheie_de_citare,*

câmp = *valoare,*

câmp = *valoare,*

⋮

}

După cum se constată, o înregistrare BIBTEX începe cu un cuvânt–cheie precedat de semnul `@`, care reprezintă **tipul** referinţei. Acest cuvânt–cheie poate fi:

`ARTICLE`	articol
`BOOK`	carte
`BOOKLET`	broşură
`CONFERENCE`	lucrările unei conferinţe
`INBOOK`	capitol dintr-o carte
`INCOLLECTION`	carte dintr-o colecţie
`INPROCEEDINGS`	lucrare de la o conferinţă
`MANUAL`	manual
`MASTERSTHESIS`	teză de master
`MISC`	diverse
`PHDTHESIS`	teză de doctorat
`PROCEEDINGS`	lucrările unei conferinţe
`TECHREPORT`	raport tehnic
`UNPUBLISHED`	lucrare nepublicată

Cheia de citare reprezintă o succesiune de litere şi cifre, unică, ce va fi folosită pentru referirea la înregistrare în cadrul unui fişier LATEX (folosind `\cite`).

Se recomandă folosirea unui sistem raţional de generare a cheilor de citare, de exemplu: iniţiala prenumelui şi numele de familie al primului autor, urmate de anul apariţiei şi de o literă sau cifră, cum ar fi: `imunteanu97a`, `imunteanu97b`, ...

Chei de citare precum `Munteanu` nu sunt recomandabile: autorul Munteanu Ion nu va scrie probabil o singură operă pe parcursul carierei sale, iar cu acest sistem Munteanu George ar trebui să aibă aceeaşi cheie de citare! De asemenea, chei de citare conţinând iniţiale precum `Bsd22m` nu sunt recomandabile deoarece semnificaţia acestei combinaţii de litere se uită după scurt timp.

După cheia de citare urmează, între acolade, o listă de **câmpuri** ale înregistrării, separate prin virgulă. Câmpurile sunt obligatorii sau opţionale. O parte dintre cele opţionale sunt predefinite, dar utilizatorul poate defini oricâte noi câmpuri opţionale.

Câmpurile **predefinite** ale unei înregistrări BIBTEX sunt:

- `KEY` (opţional) – poate conţine, de exemplu, un cod al domeniului căruia îi aparţine lucrarea, de exemplu: “matem”, “electromag”, “VLSI”, etc.
- `ADDRESS` (opţional) – adresa editorului sau a conferinţei
- `AUTHOR` (obligatoriu) – lista autorilor, separaţi prin cuvântul `and`

  Exemplu:
  `AUTHOR = {Ion Popescu and George Ionescu and Virgil Georgescu}`
- `TITLE` (obligatoriu la tipul `ARTICLE`) – titlul lucrării.

  În titlu se pot folosi comenzi LaTeX, cum ar fi `\c{a}` pentru a obţine litera ă, `$L^2$` pentru a obţine L^2, etc.

  Majoritatea stilurilor de citare LaTeXtransformă literele mari din titlu în litere mici. De exemplu, titlul

  ```
  TITLE = {Ecua\c{t}iile lui Maxwell}
  ```

  va apărea în lista bibliografică sub forma

  Ecuaţiile lui maxwell

  Dacă se doreşte ca o parte a titlului să fie păstrată în lista bibliografică aşa cum apare ea în fişierul BibTeX, acea parte trebuie inclusă între acolade:

  ```
  TITLE = {Ecua\c{t}iile lui {M}axwell}
  ```
- `BOOKTITLE` (obligatoriu la tipul `BOOK` şi `INPROCEEDINGS`) – titlul cărţii
- `YEAR` (obligatoriu) – anul apariţiei
- `MONTH` (opţional) – luna apariţiei (sau luna în care a fost scrisă lucrarea, în cazul celor nepublicate). Pot fi folosite prescurtări (în limba engleză) ale lunilor anului: `jan, feb, mar, apr, may, jun, jul, aug, sep, oct, nov, dec`.
- `JOURNAL` (obligatoriu la tipul `ARTICLE`) – revista în care a apărut lucrarea
- `VOLUME` (opţional) – numărul volumului
- `PAGES` – numerele de pagini, sub forma: `120` sau `120--140`
- `PUBLISHER` – Editura
- `EDITOR` – Numele editorilor (persoane, nu instituţii!) în cazul cărţilor cu mai mulţi autori
- `TYPE` (opţional) – Tipul raportului tehnic

Pot fi incluse şi alte câmpuri definite de utilizator, cum ar fi: `ISBN`, `ISSN`, `ABSTRACT`, etc.

4.11.2 Introducerea referinţelor bibliografice în fişierele LaTeX

Citările se fac în mod obişnuit, cu

`\cite{cheie_de_citare}`

sau

`\nocite{cheie_de_citare}`,

unde `cheie_de_citare` este una din cheile dintr-un fişier `.bib`.

În locul unde se doreşte bibliografia, este necesară introducerea, în fişierul `.tex`, a două comenzi LaTeX:

`\bibliographystyle{`*stil*`}`

`\bibliography{`*listă_fiş_bib*`}`

Stilul bibliografic

Parametrul *stil* este unul dintre **stilurile bibliografice** (a nu se confunda cu stilurile LaTeX) existente. Stilurile bibliografice sunt memorate în fişiere cu extensia `.bst`, care conţin regulile de punere în pagină a unei referinţe, de exemplu: iniţiala prenumelui urmată de numele de familie al autorului, titlul cu litere italice, numele revistei cu litere drepte, numărul volumului precedat de literele "vol.", etc.

Cele mai des utilizate fişiere de stil bibliografic sunt:

- `plain.bst` (declarat cu `\bibliographystyle{plain}`), în care citările sunt numerotate între paranteze pătrate ([1]) şi sunt sortate în ordinea alfabetică a numelor autorilor;
- `unsrt.bst`, asemănător cu `plain` dar referinţele apar în listă în ordinea în care au fost citate în text;
- `alpha.bst`, asemănător cu `plain`, dar citările sunt alfa–numerice ([Pop97] pentru o lucrare din 1997 scrisă de autorul Popescu);
- `abbrv.bst`, asemănător cu `plain`, dar cu folosirea de prescurtări pentru prenume, lună şi numele revistei;
- `plainrom.bst` şi `unsrtrom.bst`, asemănătoare cu `plain`, respectiv `unsrt`, dar adaptate documentelor scrise în limba română (aceste stiluri nu sunt standard, ci au fost realizate de I. Munteanu folosind utilitarul `makebst`, pentru a fi utilizate în LMN).

Revistele de prestigiu au propriile lor stiluri bibliografice, de exemplu:

`ieee.bst` – la revista IEEE Transactions on Magnetics

`ieeetran.bst` – la celelalte reviste IEEE

`acm.bst` – la revistele ACM (Association of Computing Machinery)

Lista de fişiere de bibliografie

Parametrul *listă_fiş_bib* este o listă de fişiere `.bib`, fără extensie şi fără blancuri între ele, separate prin virgule.

Exemplu: `\bibliography{carti,articole}`

unde `carti.bib` şi `articole.bib` sunt două fişiere conţinând înregistrări BibTeX.

Comenzi pentru procesarea fişierelor LaTeX cu includerea bibliografiei

Comanda care permite includerea listei de referinţe citate în fişierul LaTeX este `bibtex`.

O succesiune tipică de comenzi pentru realizarea fişierului imprimabil în cazul în care bibliografia este inclusă din fişiere `.bib` este (să presupunem că fişierul LaTeX are numele `raport.tex`):

```
latex raport
bibtex raport
latex raport
latex raport
dvips -t a4 -o raport.ps raport
```

Se constată că după executarea comenzii `bibtex`, este necesară compilarea de cel puţin două ori cu LaTeX, pentru ca citările să apară corect în text.

4.11.3 Fişiere utilizate în BibTeX

`.bst`	-	definiţia stilului bibliografic
`.bib`	-	conţine referinţe sub formă de înregistrări BibTeX
`.blg`	-	conţine lista eventualelor erori apărute în urma executării BibTeX
`.bbl`	-	conţine lista de referinţe citate în fişierul `.tex`, formatate conform definiţiei din fişierul de stil (`.bst`) folosit.

4.11.4 Exemple de utilizare

În cele ce urmează este prezentat rezultatul folosirii diferitelor stiluri bibliografice, pentru acelaşi set de citări.

Fişierul `.tex` utilizat a avut structura:

```
\documentstyle[12pt,a4wide,romanian]{article}

\renewcommand{\refname}{\vspace*{-1cm}}
\pagestyle{empty}

\begin{document}
\section*{Stilul \tt\bfseries plain}
\nocite{reichard92a} \nocite{wolfram96a} \nocite{munteanu96b1} \nocite{ioan92a}
\nocite{ioan92c}

\bibliographystyle{plain}
\bibliography{ref,unix,controlsys,ref1,book}     % se inlocuieste pe rand cu:
                                                  % unsrt,alpha,abbrv,unsrtrom
\end{document}
```

Rezultatul procesării cu LaTeX şi cu BIBTEX este prezentat mai jos.

Stilul `plain`

[1] D. C. Ioan and Irina Munteanu. Numerical methods in the electrical engineer's education. *European J. of Engineering Education*, 17(2):151–157, 1992.

[2] D. C. Ioan, Irina Munteanu, B. Ionescu, M. Popescu, and R. Popa. Computer aided learning of numerical methods for electrical engineers. In *3rd World Conf. on Engineering Education*, pages 15–20, Portsmouth, United Kingdom, September 20–25 1992.

[3] Irina Munteanu and F. M. G. Tomescu. Optimisation of a magnetic device based on symbolic analysis of the field. *IEEE Trans. Magn.*, 33(2):1840–1843, March 1997.

[4] K. Reichard and E. F. Johnson. *Teach Yourself UNIX*. MIS Press, New York, 1992.

[5] Stephen Wolfram. *The Mathematica Book*. Wolfram Media / Cambridge University Press, 3 edition, 1996.

Stilul `unsrt`

[1] K. Reichard and E. F. Johnson. *Teach Yourself UNIX*. MIS Press, New York, 1992.

[2] Stephen Wolfram. *The Mathematica Book*. Wolfram Media / Cambridge University Press, 3 edition, 1996.

[3] Irina Munteanu and F. M. G. Tomescu. Optimisation of a magnetic device based on symbolic analysis of the field. *IEEE Trans. Magn.*, 33(2):1840–1843, March 1997.

[4] D. C. Ioan and Irina Munteanu. Numerical methods in the electrical engineer's education. *European J. of Engineering Education*, 17(2):151–157, 1992.

[5] D. C. Ioan, Irina Munteanu, B. Ionescu, M. Popescu, and R. Popa. Computer aided learning of numerical methods for electrical engineers. In *3rd World Conf. on Engineering Education*, pages 15–20, Portsmouth, United Kingdom, September 20–25 1992.

Stilul `alpha`

[IM92] D. C. Ioan and Irina Munteanu. Numerical methods in the electrical engineer's education. *European J. of Engineering Education*, 17(2):151–157, 1992.

[IMI+92] D. C. Ioan, Irina Munteanu, B. Ionescu, M. Popescu, and R. Popa. Computer aided learning of numerical methods for electrical engineers. In *3rd World Conf. on Engineering Education*, pages 15–20, Portsmouth, United Kingdom, September 20–25 1992.

[MT97] Irina Munteanu and F. M. G. Tomescu. Optimisation of a magnetic device based on symbolic analysis of the field. *IEEE Trans. Magn.*, 33(2):1840–1843, March 1997.

[RJ92] K. Reichard and E. F. Johnson. *Teach Yourself UNIX*. MIS Press, New York, 1992.

[Wol96] Stephen Wolfram. *The Mathematica Book*. Wolfram Media / Cambridge University Press, 3 edition, 1996.

Stilul `abbrv`

[1] D. C. Ioan and I. Munteanu. Numerical methods in the electrical engineer's education. *European J. of Engineering Education*, 17(2):151–157, 1992.

[2] D. C. Ioan, I. Munteanu, B. Ionescu, M. Popescu, and R. Popa. Computer aided learning of numerical methods for electrical engineers. In *3rd World Conf. on Engineering Education*, pages 15–20, Portsmouth, United Kingdom, Sept. 20–25 1992.

[3] I. Munteanu and F. M. G. Tomescu. Optimisation of a magnetic device based on symbolic analysis of the field. *IEEE Trans. Magn.*, 33(2):1840–1843, Mar. 1997.

[4] K. Reichard and E. F. Johnson. *Teach Yourself UNIX.* MIS Press, New York, 1992.

[5] S. Wolfram. *The Mathematica Book.* Wolfram Media / Cambridge University Press, 3 edition, 1996.

Stilul `unsrtrom`

[1] K. Reichard şi E. F. Johnson. *Teach Yourself UNIX*. MIS Press, New York, 1992.

[2] Stephen Wolfram. *The Mathematica Book*. Wolfram Media / Cambridge University Press, a treia ed., 1996.

[3] Irina Munteanu şi F. M. G. Tomescu. Optimisation of a magnetic device based on symbolic analysis of the field. *IEEE Trans. Magn.*, vol. 33, nr. 2, pp. 1840–1843, mar. 1997.

[4] D. C. Ioan şi Irina Munteanu. Numerical methods in the electrical engineer's education. *European J. of Engineering Education*, vol. 17, nr. 2, pp. 151–157, 1992.

[5] D. C. Ioan, Irina Munteanu, B. Ionescu, M. Popescu şi R. Popa. Computer aided learning of numerical methods for electrical engineers. În *3rd World Conf. on Engineering Education*, pp. 15–20. Portsmouth, United Kingdom, sept. 20–25 1992.

4.12 Configurarea LATEX de către utilizator

Pentru ca LATEXsă funcţioneze fără probleme, el trebuie să "ştie" unde sunt plasate fişierele de care are nevoie. Aceste fişiere sunt de tip:

`.sty`	–	stiluri LATEX
`.tex`	–	documente LATEX(generate de utilizator)
`.pk`	–	fişiere care conţin definiţii de fonturi
`.bib`	–	fişiere care conţin liste de referinţe BIBTEX
`.bst`	–	fişiere care conţin definiţii de stiluri bibliografice

Aceste fişiere sunt căutate de LATEX şi BIBTEX, în mod implicit, în două directoare:

- directorul curent;
- directoarele predefinite de administratorul de sistem la instalarea LATEX.

Cum utilizatorul poate defini el însuşi noi stiluri LATEXşi BIBTEX, sau poate prelua de pe Internet astfel de stiluri, va trebui să le salveze în directoare proprii (unde are drept de scriere) şi nu în cele implicite. Pentru a evita existenţa în mai multe subdirectoare a aceloraşi fişiere `.sty, .bib` sau `.bst`, se recomandă:

- Crearea, în directorul home, a unui subdirector numit `latex`.
- Crearea, în subdirectorul `latex`, a trei alte directoare:
  - `sty`
  - `bst`
  - `bib`

  în care utilizatorul va salva fişierele cu extensia corespunzătoare.
- Definirea următoarelor variabile de environment:

  `TEXINPUTS` (se referă la fişierele `.sty, .tex`, etc.)

  `TEXPKS` (se referă la fişierele `.pk`)

  `BSTINPUTS` (se referă la fişierele `.bst`)

  `BIBINPUTS` (se referă la fişierele `.bib`)

Cele patru variabile de environment pot fi definite din linia de comandă, sau direct în fişierul de configurare al shell–ului (de exemplu `.cshrc` sau `.bashrc`).

Aceste variabile trebuie să conţină liste de directoare (în care se găsesc fişierele utilizate de LATEX şi de BIBTEX), separate prin `:` (două puncte).

Exemplu de definire a variabilelor de configurare, în fişierul `.cshrc`

```
setenv TEXPKS /var/texfonts/pk/localfont
setenv TEXINPUTS ".:/usr/lib/texmf/tex/latex/latex/:$HOME/latex/sty:\
/usr/lib/texmf/tex/latex/sty/:$HOME/docs/figuri/"
setenv BSTINPUTS $HOME/latex/bst
setenv BIBINPUTS $HOME/latex/bib
```

Exemplu de definire a variabilelor de configurare, în fişierul `.bashrc`

```
declare TEXPKS=/var/texfonts/pk/localfont
declare TEXINPUTS=".:/usr/lib/texmf/tex/latex/latex/:$HOME/latex/sty:\
  /usr/lib/texmf/tex/latex/sty/:$HOME/docs/figuri/"
declare BSTINPUTS=$HOME/latex/bst
declare BIBINPUTS=$HOME/latex/bib
```

4.13 Realizarea planşelor de prezentare folosind stilul `seminar`

Pentru a uşura realizarea planşelor sau transparentelor necesare unei prezentări orale a fost definit stilul LaTeX `seminar`, a cărui principală caracteristică este aceea că foloseşte corpuri de litere mai mari.

Orice document LaTeX poate fi **transformat într-o prezentare** prin următoarele trei modificări:

- se înlocuieşte stilul LaTeX folosit (`article`, `book` sau `report`) cu stilul `seminar`;
- se adaugă, după instrucţiunea LaTeX `\begin{document}`, încă o linie:
  `\begin{slide}`
- se adaugă, înainte de instrucţiunea LaTeX `\end{document}`, încă o linie:
  `\end{slide}`

Prelucrarea noului fişier cu `latex` se face în mod obişnuit.

O prezentare fără mari pretenţii poate fi astfel realizată foarte rapid.

4.13.1 Comenzi specifice stilului `seminar`

Stilul `seminar` pune însă la dispoziţie şi alte **noi comenzi şi cadruri** pentru gestionarea prezentărilor. Iată lista celor mai importante:

- `\begin{slide}`

  ⋮

  `\end{slide}`

  Cadru care defineşte o planşă de prezentare, aşezată în pagină în format landscape.

- `\begin{slide*}`

  ⋮

  `\end{slide*}`

  Cadru care defineşte o planşă de prezentare, aşezată în pagină în format portrait.

  Observaţie: într-un fişier pot apărea oricâte cadruri `slide` şi/sau `slide*`.

- `\newslide` – echivalent, în interiorul unui slide, comenzii `\newpage`.

- `\slideframe{`*chenar*`}` – comandă pentru definirea chenarului unei planşe. Dacă nu se doreşte chenar, atunci parametrul *chenar* trebuie să aibă valoarea `none`. Valoarea implicită este `plain`.

  Prin includerea, în lista de opţiuni ale comenzii `\documentstyle`, a opţiunii `fancybox`, parametrul *chenar* poate lua şi valorile: `shadow`, `double`, `oval`, `Oval`.

- `\slidesmag{`*n*`}` – mărimea fonturilor în prezentare, cu *n* între -5 şi 9.

Opţiuni ale comenzii `\documentstyle`, specifice stilului `seminar`

- `article` – planşele sunt micşorate şi tipărite câte două pe o pagină (util în faza de pregătire a planşelor)

- `portrait` – imprimare pe hârtie aşezată în format portrait.

  În mod implicit, informaţia cuprinsă în fişierul `.dvi` despre tipul hârtiei este că aceasta este aşezată în format landscape, deci pentru a imprima corect trebuie dată comanda `dvips` cu parametrul `landscape`:

  `dvips -t landscape` *fişier*

  La utilizare, în lista de opţiuni ale comenzii `\documentstyle`, a opţiunii `portrait`, se consideră hârtia aşezată vertical (portrait).

  Observaţie: această opţiune se referă numai la poziţia hârtiei, nu şi a textului în pagină (de aşezarea textului în pagină se "ocupă" `slide` – landscape şi respectiv `slide*` – portrait).

Pentru o listă completă a comenzilor, se recomandă consultarea documentaţiei [44] şi [82].

4.13.2 Exemplu de utilizare

Conclusions

- The proposed technique appears to be an efficient alternative for determining lumped circuit parameters for a class of general distributed–parameters devices.
- It can be used for extracting parameters valid over a wide range of frequencies with only one time–domain field simulation.
- Results obtained so far are promising.

Şi iată cum a fost realizat:

```
% O prezentare cu o singura plansa slide*
\documentstyle[epsf,semrot,portrait,a4,fancybox]{seminar}

% -- page style --
\pagestyle{empty}
\slideframe{shadow}

\begin{document}

\begin{slide*}
\section*{Conclusions}
\begin{itemize}
\item The proposed technique appears to be an efficient alternative for
determining lumped circuit parameters for a class of general
distributed--parameters devices.
\item It can be used for extracting parameters valid over a wide range of
frequencies with only one time--domain field simulation.

\item Results obtained so far are promising.
\end{itemize}
\end{slide*}

\end{document}
```

4.14 Folosirea programului utilitar make

În vederea "automatizării" operaţiilor necesare pentru a obţine fişierul imprimabil, plecând de la un fişier–sursă LaTeX, se poate folosi utilitarul make, descris mai pe larg în paragraful 7.3.

Pentru aceasta, în directorul în care se găsesc fişierele Dvs. LaTeX trebuie să generaţi un fişier numit Makefile, a cărui structură poate fi următoarea:

```
# Numele fisierului .tex, fara extensie
FILE = referat

$(FILE).ps: $(FILE).dvi $(FILE).aux
	dvips -t a4 $(FILE) -o $(FILE).ps

cleanall:
	rm -f *.dvi *.aux *.log

clean:
	rm -f $(FILE).dvi $(FILE).aux $(FILE).log

$(FILE).dvi: $(FILE).tex
	latex $(FILE)

ghost: $(FILE).ps
	ghostview $(FILE).ps &
```

Comenzile din acest fişier (de genul `latex ...` sau `dvips ...`) vor fi indentate folosind <Tab> şi nu spaţii.

Utilitarul se utilizează astfel:

- editaţi fişierul Makefile pentru a înlocui numele fişierului (definit prin FILE =) cu numele fişierului Dvs. LaTeX, fără extensie (să presupunem că acesta este chiar referat.tex);
- pentru a genera fişierul imprimabil daţi comanda:

  `make`
- pentru a şterge fişierele referat.aux, referat.dvi, referat.log (e bine să faceţi acest lucru după ce aţi terminat de redactat şi imprimat documentul):

  `make clean`
- pentru a şterge toate fişierele cu extensia .aux, .dvi şi .log:

  `make cleanall`
- pentru a vizualiza a fişierului imprimabil PostScript:

  `make ghost`

4.15 Exerciţii

EXERCIŢIUL 1

Creaţi structura de directoare propusă în paragraful 4.12 (directoarele `latex`, `bst`, `sty`, `bib`).

În directorul ~/`latex/bib`, creaţi fişierul `ref.bib` în care veţi include lista Dvs. de referinţe bibliografice. Pentru început, introduceţi o singură referinţă, după modelul de mai jos:

```
@BOOK{CIMocanu81a,
        AUTHOR              = {C.~I.~Mocanu},
        ADDRESS             = {Bucure\c{s}ti},
        PUBLISHER           = {Editura Didactic\u{a} \c{s}i Pedagogic\u{a}},
        TITLE               = {Teoria c\^ampului electromagnetic},
        YEAR                = {1981}
}
```

Observaţie: nu este recomandabil să folosiţi, în fişiere `.bib`, prescurtări de tipul `´a` pentru a obţine diacriticele româneşti.

EXERCIŢIUL 2

Editaţi fişierul ~/`.bashrc`, adăugând comenzile de configurare propuse în paragraful 4.12. Daţi apoi comanda

```
source ~/.bashrc
```

pentru ca noile setări să devină active.

EXERCIŢIUL 3

În directorul ~/`latex` creaţi fişierul `Makefile` cu structura descrisă în paragraful 4.14. Copiaţi apoi fişierul `Makefile` în directorul ~/`metnum`.

EXERCIŢIUL 4

În directorul `metnum` scrieţi un fişier numit `modelref.tex` care conţine elementele principale ale unui referat de laborator. (Veţi folosi acest model la toate laboratoarele de pe parcursul anului de studii aprofundate.)

O propunere de structură a fişierului `modelref.tex` este prezentată mai jos.

```
%%%%%%%%%%%%%%% MODEL DE REFERAT DE LABORATOR %%%%%%%
% Fisier: modelref.tex
% Se copiaza sub alt nume la fiecare nou referat
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

\documentstyle[11pt,a4wide,psfig,rom,romanian]{article}

% Punere in pagina
\addtolength{\topmargin}{-1.5cm}
\addtolength{\textheight}{2cm}

% Definitii si noi comenzi
\newcommand{\instructor}[1]
        {\vspace*{-4ex}\centerline{{\bf Instructor}: #1}}

\newcommand{\antet}{{\normalsize
        U.P.B. \ \ Laboratorul de Metode Numerice --- LMN}
        \vspace*{3ex}}

\title{\antet\\
\bf Titlul lucr'arii}         % Inlocuiti cu titlul real
\author{{\bf Ion Popescu}\\   % Inlocuiti cu numele Dvs. si adresa email
       {\small {\tt ipopescu@lmn.pub.ro}, Grupa 161M,
                           Facultatea Electrotehnic'a}}
\date{\today}

\begin{document}
\maketitle
\thispagestyle{empty}
\instructor{'S.l. I. Munteanu}  % Inlocuiti cu numele cadrului didactic

\tableofcontents                % Cuprinsul lucrarii
\vspace*{0.5cm}                 % Spatiu vertical; la stilul report poate lipsi

%%%% Aici incepe referatul de laborator propriu-zis
% Referatul este structurat cu sectiuni si subsectiuni
```

```
\section{Introducere}          % Numele acestei sectiuni este doar un exemplu
\section{Formularea problemei}
\section{Metoda de rezolvare}
\section{Rezultate ob´tinute}
\section{Concluzii}

\norom                         % Inhiba actiunea caracterului ´
\bibliographystyle{plain}
\bibliography{ref}             % Deocamdata foloseste doar ref.bib
\addcontentsline{toc}{section}{Bibliografie}

\rom                           % Caracterul ´ genereaza diacritice

% Referatul poate contine si anexe
\appendix
\section{Codul surs´a al programului}  % Numele sectiunii este doar un exemplu

\end{document}
```

Compilaţi fişierul `modelref.tex` cu LaTeX şi eliminaţi eventualele erori. La vizualizarea cu `xdvi modelref`, rezultatul ar trebui să fie asemănător cu figura de pe pagina 112.

EXERCIŢIUL 5

După aceste pregătiri, veţi începe acum crearea primului Dvs. "referat de laborator". (Deoarece scopul acestei lucrări este de a vă deprinde cu utilizarea LaTeX, veţi constata că o parte din secţiunile standard cuprinse în `modelref.tex` vor lipsi. Vă recomandăm ca, atunci când va trebui să scrieţi un adevărat referat de laborator să utilizaţi structura completă.)

Pentru siguranţă, copiaţi modelul `modelref.tex` în directorul `~/latex`.

Copiaţi modelul `modelref.tex` sub numele `rc.tex` (evident, în directorul `metnum`).

Modificaţi fişierul `rc.tex` astfel încât documentul să reproducă identic (cu excepţia numelor şi datei) modelul de pe pagina 112.

Notă: în realizarea modelului a fost folosită dimensiunea de 10pt a caracterelor.

Observaţie: NU scrieţi niciodată numărul ecuaţiei sau al referinţei direct, ci folosiţi comenzile de referire `\ref{ }`, respectiv de citare `\cite{ }`.

U.P.B. Laboratorul de Metode Numerice — LMN

Titlul lucrării

Ion Popescu
`ipopescu@lmn.pub.ro`, Grupa 161M, Facultatea Electrotehnică

28 iulie 1998

Instructor: Ş.l. I. Munteanu

Cuprins

1 Introducere 1

2 Formularea problemei 1

3 Metoda de rezolvare 1

4 Rezultate obţinute 1

5 Concluzii 1

Bibliografie 1

A Codul sursă al programului 1

1 Introducere

2 Formularea problemei

3 Metoda de rezolvare

4 Rezultate obţinute

5 Concluzii

A Codul sursă al programului

U.P.B. Laboratorul de Metode Numerice — LMN

Studiul unui circuit RC serie în regim tranzitoriu

Ion Popescu
ipopescu@lmn.pub.ro, Grupa 161M, Facultatea Electrotehnică

19 noiembrie 1997

Instructor: Ş.l. I. Munteanu

Cuprins

1 Ecuaţiile de funcţionare ale rezistorului şi condensatorului 1
1.1 Ecuaţia de funcţionare a rezistorului ideal 1
1.2 Ecuaţia de funcţionare a condensatorului ideal 1

Bibliografie 2

1 Ecuaţiile de funcţionare ale rezistorului şi condensatorului

1.1 Ecuaţia de funcţionare a rezistorului ideal

Fie un fir conductor C dintr-un material izotrop, liniar şi omogen, de conductivitate σ, parcurs de curent cu intensitatea i. Firul are lungimea l şi aria secţiunii A, cu l mult mai mare decât dimensiunile secţiunii.

Tensiunea electrică u de-a lungul firului şi intensitatea i prin orice secţiune S sunt prin definiţie:

$$u \stackrel{D}{=} \int_C \mathbf{E} \cdot \mathbf{ds}; \qquad i \stackrel{D}{=} \int_S \mathbf{J} \cdot \mathbf{n} \mathrm{d}\, A \tag{1}$$

Conform legii conducţiei electrice, în orice punct din conductor

$$\mathbf{E} = \rho \mathbf{J}, \tag{2}$$

cu $\rho = 1/\sigma$. Dacă se consideră densitatea de curent uniform repartizată pe orice secţiune ($J = i/A$), iar $\mathbf{E}$ ($\mathbf{J}$) orientat de-a lungul conductorului, atunci relaţiile (1) şi (2) conduc la:

$$u = \int_C \rho \mathbf{J} \cdot \mathbf{ds} = \rho J l = \frac{\rho l}{A} i = Ri. \tag{3}$$

1.2 Ecuaţia de funcţionare a condensatorului ideal

Ecuaţia de funcţionare a condensatorului ideal poate fi dedusă folosind teorema conservării sarcinii electrice şi proporţionalitatea dintre sarcina electrică q a unei armături şi tensiunea electrică u dintre armături. Această ecuaţie are forma [1]:

$$i = C \frac{\mathrm{d} u}{\mathrm{d} t}, \tag{4}$$

unde C este capacitatea condensatorului, care nu depinde de sarcină şi de tensiunea la borne, ci numai de geometria condensatorului şi de permitivitatea ε a acestuia.

Bibliografie

[1] C. I. Mocanu. *Teoria câmpului electromagnetic.* Editura Didactică şi Pedagogică, Bucureşti, 1981.

EXERCIŢIUL 6

Adăugaţi o nouă secţiune la referatul de laborator, care să reproducă modelul de mai jos. Completaţi formulele corecte ale ecuaţiilor (5) şi (6).

2 Ecuaţia tensiunii la bornele condensatorului

Fie un circuit RC serie, alimentat cu o sursă de tensiune electromotoare constantă E. Un contact K înseriat cu circuitul se închide la momentul $t = 0$, când tensiunea la bornele condensatorului este u_{C_0}.

A doua ecuaţie Kirchhoff are forma:

$$\sum_{k \in [o]} u_k = 0; \qquad \text{sau} \qquad u_R + u_C = E.$$

Folosind ecuaţiile de funcţionare (3) şi (4) ale rezistorului şi condensatorului, relaţia devine:

Completaţi Dvs. (5)

Soluţia ecuaţiei diferenţiale (5) are forma:

Completaţi Dvs. (6)

EXERCIŢIUL 7

Adăugaţi încă o secţiune la referatul de laborator, numită **Comparaţie între circuitele RC serie şi RL serie**, care să conţină următorul tabel, centrat orizontal şi numerotat automat de LaTeX:

Tip	Variabilă de stare	Ecuaţie
R–L	i (i_L)	$L\frac{di}{dt} + Ri - E = 0$
R–C	u (u_C)	$RC\frac{du}{dt} + u - E = 0$

Tabelul 1: Comparaţie

Indicaţie. Includeţi un cadru `tabular` într-un cadru `table`.

EXERCIŢIUL 8

Examinaţi documentul LaTeX de pe pagina 116.

Cum se pot realiza: • mărirea marginii din dreapta a paginii; • litera Σ de dimensiune mare; • alinierea la stânga a textului în coloana I a tabelului; • linia verticală lungă în ecuaţie; • indicele S_N din ecuaţie?

Ecuatia Poisson generalizata

Nume Autor

19 octombrie 1997

Cuprins

1 Formularea problemei **1**
1.1 Ecuatia de rezolvat 1
2 Conditii de frontiera **1**

1 Formularea problemei

1.1 Ecuatia de rezolvat

Ne propunem sa rezolvam o ecuatie de tip Poisson generalizata, care are forma:

$$div\,(K\,gradV) = f \tag{1}$$

in cazul unor domenii bidimensionale, intr-un domeniu de studiu D marginit de frontiera $\sum$.

Ecuatia (1) acopera o categorie larga de probleme de interes in inginerie. In tabelul de mai jos sunt prezentate semnificatia necunoscutei V precum si expresiile pe care le au constanta de material K si sursa F in cateva cazuri de interes:

Regimul	Necunoscuta V	Constanta de material K	Sursa f
Electrostatic	V	ε	$-\rho_v + div\mathbf{P}_p$
Electrocinetic	V	σ	$div\mathbf{J}_i$

2 Conditii de frontiera

Vom considera ca pe portiunea S_N a frontierei domeniului se cunosc valorile derivatei dupa normala a potentialului:

$$\left.\frac{dV}{dn}\right|_{S_N} = f_N.$$

⋮

Exerciţiul următor este facultativ.

EXERCIŢIUL 9

Adăugaţi un index la referatul de laborator.

Pentru aceasta:

- Adăugaţi `makeidx` în lista de opţiuni ale comenzii `\documentstyle`.
- Adăugaţi comanda

  `\makeindex`

  în fişierul `.tex`, înainte de `\begin{document}`.
- Adăugaţi comanda

  `\printindex`

  în fişier, în locul unde doriţi să apară indexul (de obicei la sfârşitul documentului, după bibliografie).
- Marcaţi în textul documentului intrările din index, după modelele:

  `. . . densitatea de curent \index{Densitate de curent}`

  (cuvintele *Densitate de curent* vor apărea în index, urmate numărul paginii unde sunt plasate cuvintele `densitatea de curent`).

  `. . . Aceast´a ecua´tie are forma \index{Ecua´tii de func´tionare!condensator}`

  (astfel se defineşte un index pe două nivele, cuvântul *condensator* va apărea ca nivel inferior al intrării din index *Ecuaţii de funcţionare*).
- Procesaţi fişierul pentru a obţine forma sa imprimabilă:

```
latex rc
makeindex rc
latex rc
dvips -t a4 -o rc.ps rc
```

Observaţii.

Realizarea indexului este una din **cele mai anevoioase** activităţi în timpul pregătirii unui document. Este bine să vă uşuraţi munca, scriind câteva macrouri speciale LaTeX şi `vi` (sau `emacs`).

De exemplu, indexul cărţii de faţă a fost realizat astfel:

1. Au fost definite **trei noi comenzi** LaTeX (preluate din [53], pag. 352):

- `\Index`, care permite introducerea unui text atât în document cât şi în index; în acest fel se evită tastarea de două ori a aceluiaşi text;
- `\Indextt`, care funcţionează la fel ca `\Index`, dar fontul textului este Courier (`\tt`) atât în document cât şi în index;
- `\Com`, care permite introducerea în document şi în index a numelui unei comenzi (care începe cu caracterul \).

Noile comenzi LaTeX au fost definite astfel:

```
\newcommand{\Index}[1]{#1\index{#1}}
\newcommand{\Indextt}[1]{{\tt #1}\index{#1@{\tt{#1}}}}
\newcommand{\bs}{\symbol{'134}}
\newcommand{\Com}[1]{{\tt {\bs#1}}\index{#1@{\tt {\bs#1}}}}
```

2. Au fost adăugate **două noi macrouri** în fişierul de configurare a editorului `vi` (simbolul ↑ semnifică tasta <Ctrl>):
   - ↑`a` – aplică macroul LaTeX `\Index` unui cuvânt care trebuie să apară în index; astfel, dacă iniţial cuvântul se afla în fişierul `.tex` sub forma
     `cuvant`
     după ce se plasează cursorul pe începutul său şi se dă comanda ↑`a`, el se modifică astfel:
     `\Index{cuvant}`
   - ↑`z` – aplică macroul LaTeX `\Indextt` unui cuvânt.

   Noile macrouri `vi` au fost adăugate în fişierul `.exrc` sub forma:

   ```
   map ↑v↑a i<Bslash>Index↑v↑[ea↑v↑[
   map ↑v↑a i<Bslash>Indextt↑v↑[ea↑v↑[
   ```

   (↑`v` permite introducerea ca atare în fişier a unui caracter de control, cum este ↑`a`, iar ↑`[` are acelaşi efect ca şi tasta <ESC>.)

3. A fost definit **un nou fişier de stil pentru index** (inspirat tot din [53]), numit `mystyle.ist`. (Extensia `.ist` este cea implicită pentru fişierele de stil al indexului.)

   Structura fişierului este următoarea:

   ```
   % Fisier de stil pentru MakeIndex -- mystyle.ist
   %--- Litera inaintea fiecarui grup de intrari din index
   heading_prefix "{\\bfseries\\hfil "      % Insereaza in fata literei
   heading_suffix "\\hfil}\\nopagebreak\n"  % Adauga dupa litera
   headings_flag  1                         % Pune antet (litere mari)
   %--- Delimitatori (linie punctata)
   delim_0   "\\dotfill "
   delim_1   "\\dotfill "
   delim_2   "\\dotfill "
   ```

4. Pentru a folosi noul fişier de stil, comanda de generare a indexului a fost (numele fişierului `.tex` fiind `carte.tex`):

```
makeindex -s mystyle.ist carte
```

Capitolul 5

Grafice şi figuri — Gnuplot, XFig, XPaint, XV —

Spre deosebire de literatura beletristică, orientată aproape exclusiv pe text, literatura ştiinţifică este caracterizată prin bogăţia de reprezentări grafice pe care le conţine. Acestea pot fi clasificate în:

- **grafice**
  - de una sau mai multe funcţii de o variabilă
  - funcţii de două variabile, suprafeţe tridimensionale (3D)
  - alte tipuri de reprezentări ale datelor ştiinţifice cum sunt: contururi echipotenţiale, spectre ale liniilor de câmp, etc.
- **desene** şi imagini de sinteză
  - scheme bazate pe simboluri convenţionale – abstracte (circuite, diagrame de stare, organigrame, scheme logice sau de fluenţă, etc.)
  - desene ale unor obiecte (desene tehnice sau artistice, cu reprezentări 2D sau 3D, cu sau fără efecte de realism de tip lumini–umbre)
  - picturi (reprezentări concrete sau abstracte în care culoarea sau textura primează faţă de linii)
- **fotografii** care reprezintă imagini reale.

Imaginile permit pe de o parte vizualizarea datelor ştiinţifice obţinute în urma experimentelor sau din calcul şi pe de altă parte reprezentarea obiectelor reale sau virtuale. Utilizarea calculatorului în activitatea ştiinţifică presupune, în consecinţă, folosirea unor programe care să permită manipularea tuturor acestor tipuri de reprezentări grafice [42]. Aceste pachete au următoarele funcţii tipice:

- **crearea**
- **memorarea**
- **editarea**:
  - transformarea (măriri, micşorări, translatarea, rotire)
  - ştergerea unor părţi sau elemente
  - copierea unor elemente, eventual repetată
  - inserarea şi suprapunerea (inclusiv adnotarea)
  - modificarea culorilor sau a contrastului
- **inserarea** în documente sau producţii
  - vizualizare pe ecran de calculator sau TV
  - imprimare pe hârtie sau pe alt suport (de ex. tranparent sau film).

În general, cantitatea de informaţie necesară reprezentării unei imagini este mult mai mare decât cea necesară textului. În consecinţă, memorarea imaginilor presupune adoptarea unor tehnici eficiente, care să conducă la compresia datelor şi obţinerea unor fişiere de dimensiuni cât mai mici.

Se deosebesc trei categorii fundamentale de astfel de tehnici:

- reprezentarea **vectorială**, specifică desenelor, în care se memorează secvenţe de puncte, linii, curbe, texte, etc.;
- reprezentarea pixel (“**bitmaps**”), specifică picturilor şi fotografiilor, în care se memorează culoarea (sau nuanţa de gri) a fiecărei celule numită pixel din care este alcătuit rastrul imaginii;
- reprezentarea metafişier (“**metafile**”), specifică imaginilor complicate, numite *scene* şi care presupune utilizarea tehnicilor vectoriale şi bitmap în acelaşi fişier.

Diversitatea tipurilor de abordare şi a pachetelor grafice a condus la apariţia unei multitudini de formate pentru fişierele grafice, dintre care cele mai importante sunt [67]:

Format		**Tip**	**Extensie**	**Iniţiator**
DXF	–Drawing eXchange Format	vectorial	`ADI`	AutoDesk/AutoCAD
FIG	– Figures	vectorial	`FIG`	XFig
HPGL	–Hewlett Packard Graphic Language	vectorial	`HPG`	Hewlett Packard
GIF	–Graphic Interchange Format	bitmap	`GIF`	Compuserve

TIFF	–Tag Image File Format	bitmap	TIF	Aldus
PCX	–PC Paintbrush Format	bitmap	PCX	MS Paint
BMP	–Windows Bitmap	bitmap	BMP	MS Windows
XBM	–Xwindow bitmap - mono	bitmap	XBM	Xwindow
XPM	–Xwindow Pixel Map - color	bitmap	XPM	Xwindow
PBM	–Portable Bitmap cu variantele PGM (gri), PPM (color)	bitmap	PBM, PGM, PPM	pbmplus
JPEG		bitmap	JPG	Joint Photographic Experts Group
PostScript		meta	PS, EPS	Adobe
PDF	–Portable Document Format	meta	PDF	Adobe
WMF	–Windows Meta Format	meta	WMF	MS Windows
POV	–Ray	meta	POV	Persistence of Vision Raytracer
VRML	–Virtual Reality Markup Language	meta	VRM	Virtual Reality

Diversitatea formatelor grafice face extrem de utilă funcţia de **conversie a formatelor**, realizată de multe programe prin operaţii de import/export fişiere în formatul dorit. Operaţia de conversie nu dă întotdeauna rezultate satisfăcătoare, mai ales atunci când se modifică tipul de reprezentare, motiv pentru care se recomandă memorarea imaginilor într-un format grafic corespunzător: vectorial pentru grafice, desene şi scheme şi respectiv bitmap pentru picturi şi fotografii.

Dintre pachetele de programe care permit generarea şi prelucrarea imaginilor, următoarele sunt disponibile în LMN:

Nume program	**Funcţionalitate**	**Formate de fişiere acceptate**	**Citare bibliografică**
`gnuplot`	permite realizarea de grafice 2D şi 3D de funcţii	DXF, FIG, HPGL, PBM şi EPS	[7]
`SciLab`	permite realizarea de grafice 2D şi 3D şi vizualizări câmpuri 2D	FIG, PS	[22]
`XFig`	permite crearea şi editarea de scheme şi desene	FIG, EPS, XBM,	[29]

Nume program	**Funcţionalitate**	**Formate de fişiere acceptate**	**Citare bibliografică**
`AutoCAD`	permite crearea şi editarea de desene	DXF	[39]
`mnm` - Midnight Modeller	permite realizarea desenelor 3D	DXF, POV	[13]
`Scen` - Scene Editor	permite realizarea de scene realiste	VRML	[20]
`gs` - ghostscript	permite conversia din PS sau PDF în diverse formate bitmap	PS, PDF/TIFF, BMP, PCX, PBM, etc.	[5]
`gv` - ghostview	permite vizualizarea fişierelor PS sau PDF	PS, PDF	[3]
`xv`	permite capturi de ecrane, transformări, modificări şi conversii de fişiere bitmap	GIF, JPEG, TIFF, PBM, PGM, PPM, XPM, PM, PCX şi EPS (la ieşire)	[31]
`XPaint`	permite editarea picturilor reprezentate bitmap	XBM, XPM, PPM, GIF, TIFF şi EPS (numai la ieşire)	[30]
`GIMP` - GNU Image Manipulation Program	program complex de editare grafică de tip Photoshop ce permite compoziţia şi retuşarea fotografică	GIF, JPEG, PCX, TIFF, XPM şi PS	[4]
`Star Draw` `Star Image`	programe de editare grafică sub Xwindow ce emulează MSDraw şi MSPaint de sub Windows		[23]
`Presto!`	program sub MS Windows ce permite scanarea imaginilor	TIFF, JPEG, PCX, BMP	

Obiectivele capitolului constau în:

- familiarizarea cu prinicpalele concepte ale editării grafice: formate de fişiere grafice, tipuri de imgini, obiecte grafice şi transformări ale acestora, etc.
- căpătarea deprinderilor minimale necesare realizării şi modificării de grafice, scheme, desene sau fotografii şi inserarea lor ca figuri în documente.

5.1 XFig

5.1.1 Apelare

- În laboratorul EB212: `xfig <↵ >`
- În laboratorul EB213: cu butonul din stânga al mouse-ului - Graphics - Xfig sau cu comanda: `/comun/bin/GetXFig <↵ >`

În sala EB213 este instalat pe maşina `lmnsun` dar poate fi rulat şi de pe alte maşini.

5.1.2 Obiecte XFig

XFig lucrează cu *obiecte grafice*, care pot fi obiecte *primitive* sau *compuse*.

Obiectele primitive

- arc
- cerc
- spline inchis
- elipsă
- linie poligonală
- poligon
- fişier EPS (Encapsulated PostScript)
- dreptunghi
- dreptunghi cu colturile rotunjite
- spline
- text

Obiecte compuse

- combinaţii de obiecte primitive;
  obiectele componente nu pot fi modificate individual.

Operaţii cu obiecte

- primitive:
  - mutare, rotire, reflectare (vertical, orizontal), scalare, copiere, aliniere (în cadrul unui obiect compus), ştergere

    Excepţii: obiectele text nu pot fi reflectate; dacă sunt rotite, acest lucru nu se vede pe ecran dar, dacă se salvează figura în format PostScript (vezi secţiunea 5.1.4), textul va apărea rotit.

    Observaţie: Datorită unui bug existent, se pare, în unele versiuni `xfig`, comanda de rotire a textului poate produce setarea în mod eronat a dimensiunilor figurii la valori enorme, ceea ce face imposibilă includerea figurii în documente. Evitaţi această manevră, pe cât posibil.
- compuse:
  - mutare, rotire, reflectare, scalare, copiere, ştergere

    Excepţii: dacă conţin dreptunghiuri, nu se pot roti decât cu multipli de 90 grade.

5.1.3 Regiuni ale ecranului

Ecranul programului XFig, prezentat în figura 5.1, conţine următoarele regiuni:

- regiunea de comandă (meniuri)
- rigle (sus, lateral)
- butoane pentru desenare
- butoane pentru editare
- indicatorul funcţiilor mouse-ului
- zona desenului propriu-zis ("canava")
- zona de setare a unor opţiuni: linii, culori, etc.

Regiunea de comandă conţine meniurile:

Quit Undo Redraw Paste File Export Print

Zona de setare opţiuni ("indicator") permite efectuarea operaţiilor:

- zoom (lupă);

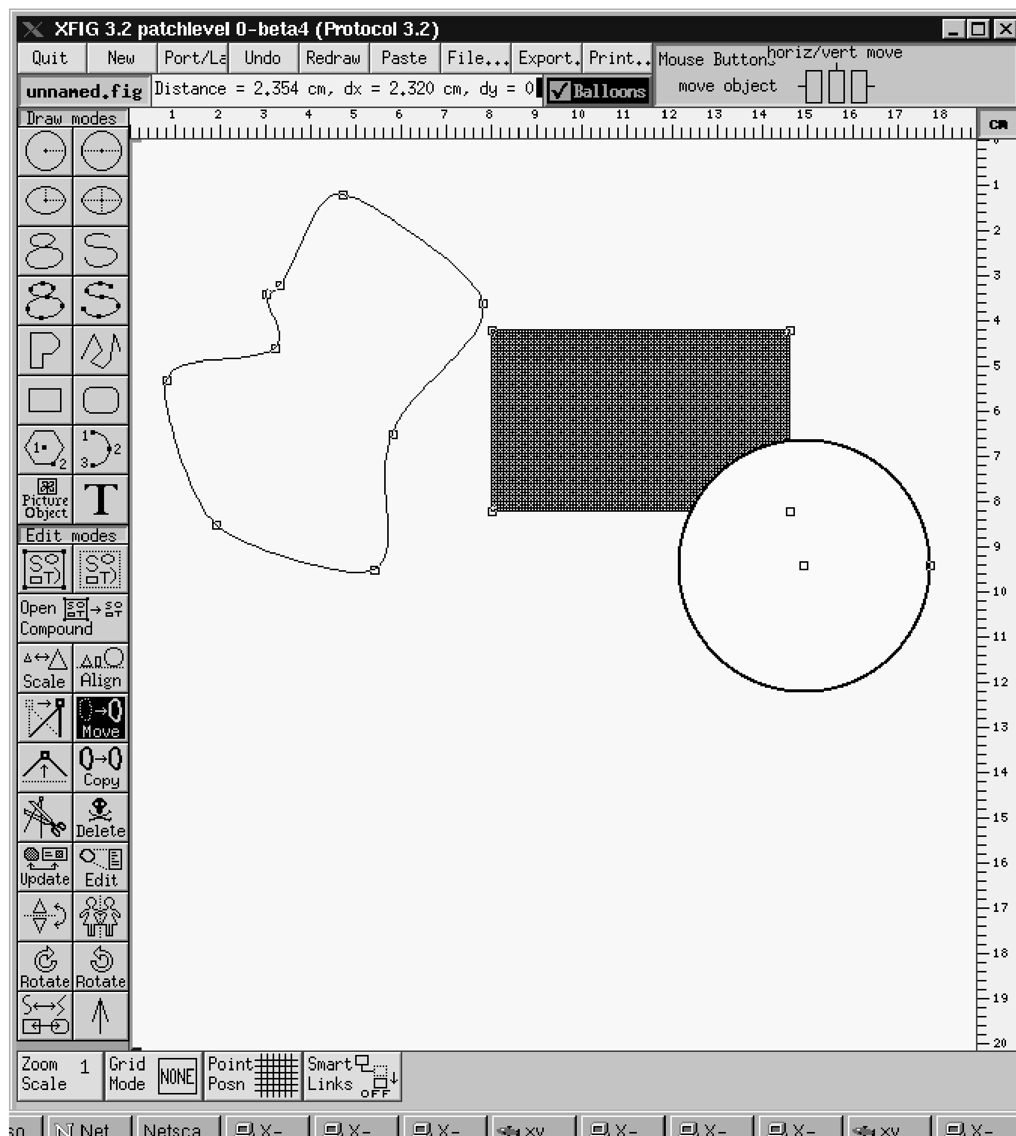


Fig. 5.1: Ecranul programului XFig

- grid (afişarea sau nu a unor linii de ghidare);
- coarseness (fineţe);
- rotation angle (unghi de rotaţie).

Zona de indicare a funcţiilor mouse-ului:

- este plasată în zona din dreapta sus a ecranului;
- sunt afişate:

  în stânga: acţiuni posibile pentru butonul din stânga;

deasupra: acţiuni posibile pentru butonul din mijloc;

în dreapta: acţiuni posibile pentru butonul din dreapta.

5.1.4 Salvarea desenului în alte formate

XFig oferă posibilitatea salvării desenului şi în alte formate decât formatul său propriu. Aceasta este o opţiune extrem de utilă, care permite utilizarea desenului cu alt program de grafică, sau permite includerea acestuia într-un document.

La salvarea unei figuri, se crează un fişier în format XFig, cu extensia implicită `.fig`. Salvarea în alt format se face selectând opţiunea `Export` din regiunea de comandă. Se pot seta:

- aşezarea paginii (Portrait, Landscape);
- formatul fişierului de ieşire (Encapsulated PostScript, de exemplu, este util pentru includerea figurii într-un fişier LaTeX; opţiunea `Latex` funcţionează bine numai pentru forme simple);
- numele noului fişier;
- directorul în care va fi creat noul fişier.

5.2 Gnuplot

Funcţionează pe toate maşinile din LMN, cu excepţia `lmndec`. În sala EB213 este instalat pe masina `sun`.

5.2.1 Apelare

- în sala EB212: din meniurile plasate pe butonul din stânga sau cu comanda `gnuplot <↵ >`
- în sala EB213: butonul din mijloc al mouse-ului - Mathematics - GNUplot sau `/comun/bin/GetPlot <↵ >`

5.2.2 Comenzi utile pentru început

`help` – lista de comenzi `gnuplot`;

`quit` sau `exit` – ieşirea din program.

5.2.3 Manualul pentru `gnuplot`

- este disponibil în sala EB212:

  `man gnuplot`

5.2.4 Ce poate face `gnuplot`

- **Trasare de grafice** pentru una sau mai multe funcţii simultan, funcţii care pot fi:
  - definite analitic în stilul limbajului C, cu ajutorul unor: operatori C, funcţii din biblioteca C, operatori specifici `gnuplot`
  - definite prin date, prin citirea unor fişiere ASCII de date
- **Alte facilităţi**:
  - definirea intervalelor de plotare pe Ox şi pe Oy
  - scalarea automată a axelor
  - marcarea axelor în diferite moduri
  - etichete pentru axe
  - funcţii şi variabile definite de utilizator
  - citirea / scrierea unui fişier de comenzi `gnuplot`

  - redirectarea ieşirii
  - calcule în complex
  - setare opţiuni XWindow

5.2.5 Comenzi gnuplot

După lansarea programului, se afişează un antet cu informaţii despre program,

se intră într-o fereastră dedicată, apare promptul `gnuplot>` şi se pot lansa comenzi `gnuplot`.

Comenzile au următoarele caracteristici:

- sunt "case-sensitive"
- pot fi abreviate
- pot apărea mai multe comenzi pe o linie, separate prin ;
- şirurile de caractere din cadrul unei comenzi sunt incluse între " sau ´
- o comandă lungă se poate extinde pe mai multe rânduri, punând câte un \ la sfârşitul unui rând

În cele ce urmează sunt descrise principalele comenzi `gnuplot`. Se vor folosi următoarele convenţii:

- între acolade { }: parametri opţionali
- cu litere drepte: cuvintele-cheie `gnuplot`
- cu litere italice: valori sau nume introduse de utilizator.

5.2.6 Comenzi de bază: plot, splot

Comenzile `plot` şi `splot` permit realizarea graficelor 2D, respectiv 3D.

Structura lor:

- Grafic funcţie definită analitic

  `plot` { `domeniu` } *funcţie* {*titlu*} {*stil*}

- Grafic funcţie definită prin date numerice

  `plot` {`domeniu`} {*"fişier_date"* {`using ...`}} {`titlu`} {*stil*}

Principalul parametru este fie funcţia, fie numele unui fişier de date între ghilimele, fie orice combinaţie a lor. **Virgula** separă grupuri de parametri corespunzători unor funcţii diferite.

Ex: `plot sin(x)`

`plot sin(x), cos(x)`

La rândul său, o funcţie poate fi:

- o expresie matematică
- o pereche de expresii matematice pentru funcţii parametrice 2D
- un triplet de expresii matematice pentru funcţii parametrice 3D

În cele ce urmează sunt descrişi parametrii şi opţiunile comenzilor `plot` şi `splot`.

Parametrul *fişier_date*

Fişierele de date au următoarea structură:

- Conţin câte o linie pentru fiecare punct de afişat:
  - 2 valori = x, y (2D)
  - 3 valori = x, y, z (3D)
  - opţional pentru grafice de erori:
    * încă o valoare ydelta = eroarea absolută a lui y
    * încă două valori ylow, yhigh = limitele inferioară şi superioară ale lui y
- Valorile sunt separate în fişier prin blancuri sau tab-uri. La grafice 2D coloana x poate lipsi, la 3D coloanele x şi y pot lipsi; în aceste cazuri, li se dau automat valori 0, 1, 2, ...
- O linie liberă în fişier va produce o întrerupere în grafic!

Opţiunea `using`

- Opţiunea `using` este folosită numai dacă graficul se face prin citirea unui fişier de date numerice.
- Poate avea ca parametru:

  `ycol`

```
xcol:ycol
xcol:ycol:ydelta
xcol:ycol:ylow:yhigh
xcol:ycol:ylow:yhigh:boxwidth
```

În cazul comenzii `splot` mai apare şi `zcol`.

`col` reprezintă coloana din fişierul de date care va fi considerată drept coloana x, y, z, eroare (ydelta), margini inferioară / superioară, dimensiunea "cutiei" pentru tipul de grafic `boxes`.

Excepţii:

– în cazul 2D, dacă apare <ycol> singur, se consideră că prima coloană este coloana x–ilor
– în cazul 3D, dacă apare <zcol> singur, nu se va citi decât acea coloană
– dacă *xcol* este 0, se va plota y în funcţie de numărul său de ordine 0, 1, 2, ...

Opţiunea `[domeniu]`

- Are sintaxa:

  `[ {var = } {xmin: xmax} ] {[ymin: ymax]}`

  unde `var` este un alt nume pentru variabila independentă.

  Când se utilizează `ranges`, scalarea automată este dezactivată. Se poate utiliza

  `[ ]` (domeniu nul)

  pentru a dezactiva scalarea automată.

  Ex: `plot sin(x)` - domeniul curent
  `plot [-10:30] sin(x)` - x între -10 şi 30
  `plot [t=-10:30] sin (t)` - alt nume pentru variabila independentă
  `plot [-pi:pi] [-3:3] tan(x), 1/x`
  `plot [:200][-pi:] exp(sin(x))` - nu se precizează decât una din limite

Opţiunea `stil`

- Are sintaxa:

  `with <stil> {<tip linie> <tip punct>},`

  unde <`stil`> poate fi (figura 5.2):

  – lines
  – points

- linespoints
- impulses
- dots
- errorbars
- steps
- boxes
- boxerrorbars

- Stilul implicit este:
  - `lines` – la grafice de funcţii definite analitic
  - `points` – la grafice de funcţii definite prin date

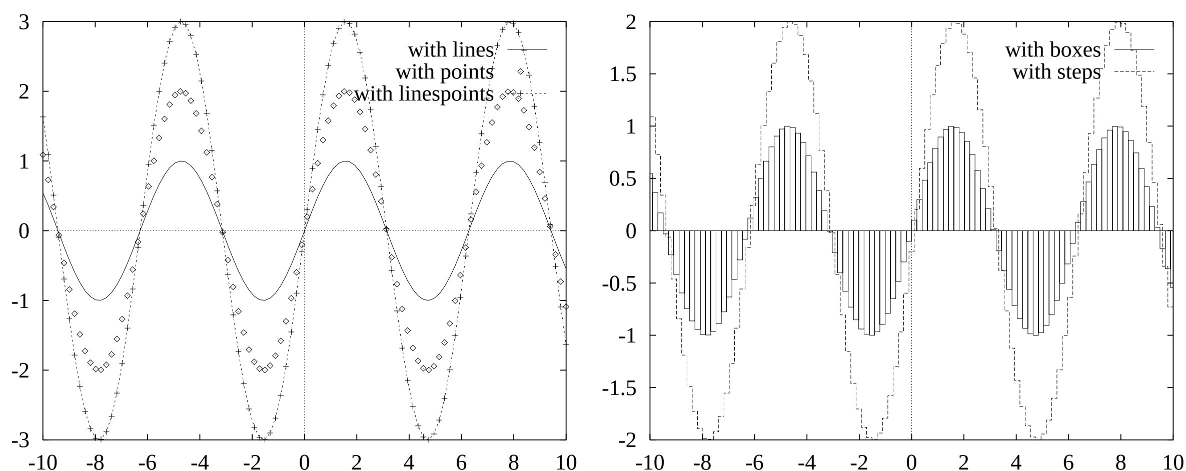


Fig. 5.2: Diferite stiluri de grafice `gnuplot`

Opţiunea `title`

- are rolul de a schimba "eticheta" unui grafic.

  Ex: `plot sin(x) title "sinus"`

5.2.7 Alte comenzi utile

- comenzi Unix:

  `cd`

  `pwd`

  `` `...` `` – substituirea cu rezultatul comenzii aflate între apostroafele inverse

- lista funcţiilor de tip C disponibile se poate obţine cu:

  `help expressions` şi selectare `functions`

- lista operatorilor disponibili se poate obţine cu:

  `help expressions` şi selectare `operators`

- lista comenzilor editorului orientat pe linie al `gnuplot`:

  `help line-editing`

- `save` – salvează în fişier:
  - funcţiile definite de utilizator
  - variabilele
  - opţiunile setate
  - ultima comandă `plot` sau `splot`

  Ex: `save set var functions "work"`

  Salvează în fişierul numit `work` opţiunile setate, variabilele şi funcţiile definite de utilizator.

- `load` – citeşte de pe disc un fişier care conţine comenzi `gnuplot`.

  Fişierul poate fi unul creat printr-o comandă `save` anterioară, sau scris de utilizator. Efectul comenzii este citirea comenzilor din fişier, linie cu linie, ca şi cum ar fi introduse de la consolă.

  Ex: `load "work"`

- lista de erori `gnuplot` cunoscute:

  `help bugs`

- numere complexe: au forma {re, im}

  Ex: `{1, 2}` pentru 1 + 2j

  Acoladele sunt obligatorii.

- `pause` – pauză, eventual cu afişarea unui mesaj

  Util în fişierele de comenzi. Sintaxa:

  `pause timp ["mesaj"]`

  unde `timp` = constantă întreagă; valori particulare:

  -1: aşteaptă CR

  0: nu aşteaptă de loc (afişează mesajul şi continuă)

  >0: aşteaptă numărul respectiv de secunde

- `reread` – citeşte de la început un fişier de comenzi. Astfel se pot crea bucle infinite.

- `set` – multe lucruri:
  - opţiuni de plotare: margine, scalare automată sau nu a axelor, etc.
  - tip axe
  - etichete axe, ... etc.
- `show parametru_set` – afişează valoarea pentru parametrul respectiv

  `show all` – afişează toate setările
- fişier util: **$HOME/.gnuplot**

  poate conţine comenzi `gnuplot` care să se execute la fiecare lansare în execuţie a programului.
- variabile şi funcţii definite de utilizator:

  `var = const`

  `funct(par1 {, par2 ...}) = expresie`

  unde numărul de parametri este mai mic sau egal cu 5.

  Ex:
  ```
  a = 5               # definirea variabilei a
  f(x) = a * sin(x)   # definirea functiei f
  ```

5.2.8 Salvarea graficelor în alte formate

Pentru a salva graficele `gnuplot` în format **PostScript** este necesară:

- Setarea tipului de terminal ca fiind "Encapsulated PostScript"

  Ex: `set term postscript eps mono "Times-Roman" 24`

  Defineşte terminalul de ieşire ca fiind de tip PostScript, cu opţiunile: Encapsulated, monocrom, font Times-Roman cu dimensiunea caracterelor de 24pt.
- Setarea ieşirii comenzilor `plot` spre un fişier:

  `set output "fis.eps"`

Pe lângă formatul PostScript, `gnuplot` permite salvarea graficelor în diferite alte formate, printre care: `latex`, `pbm` (convertibil apoi cu ajutorul programului `pbmplus` în format GIFF, PCX, etc.), `table` (tablou de valori ASCII). O listă completă a formatelor se poate obţine cu

```
help set term
```

Iniţial (la lansarea `gnuplot`) terminalul este de tip `X11`, iar ieşirea este "STDOUT". Pentru revenirea la aceste setări, se folosesc comenzile:

```
set term X11

set output
```

5.2.9 Fişiere de comenzi gnuplot

Comenzile `gnuplot` folosite pentru realizarea unui grafic pot fi scrise într-un fişier (se recomandă folosirea unei extensii specifice, de exemplu `.gnu`) şi încărcate în `gnuplot` cu ajutorul comenzii `load`.

Comenzile lungi pot fi scrise pe două sau mai multe rânduri, caracterul de continuare fiind \ (backslash). Pentru comentarii se foloseşte caracterul `#`.

Exemplu de fişier de comenzi:

```
# Grafic in scara logaritmica, o functie analitica si doua
#       definite prin date in fisierul "grafic.dat"

set logscale y 10
set grid
set function style linesp
set data style linesp
set samples 20      # pentru functii definite analitic
set key 2, 500      # plasarea etichetelor graficelor la x = 2, y = 500
                    # graficul se extinde pe x intre 0.5 si 5,
                    # iar pe y intre 0.01 si 1000
set title "Rezultatele analitice si experimentale"
set xlabel "x [m]"
set ylabel "f(x) [-]"

set term postscript eps mono "Times-Roman" 20
set output "graficx.eps"

f(x) = x**2
plot [0.1:5] \
     f(x) title "Analitic", \
    "grafic.dat" using 1:2 title "Experiment 1", \
    "grafic.dat" using 1:3 title "Experiment 2"

!mv "graficx.eps" "../figuri/"
```

Rezultatul executării acestui fişier de comenzi este prezentat în figura 5.3

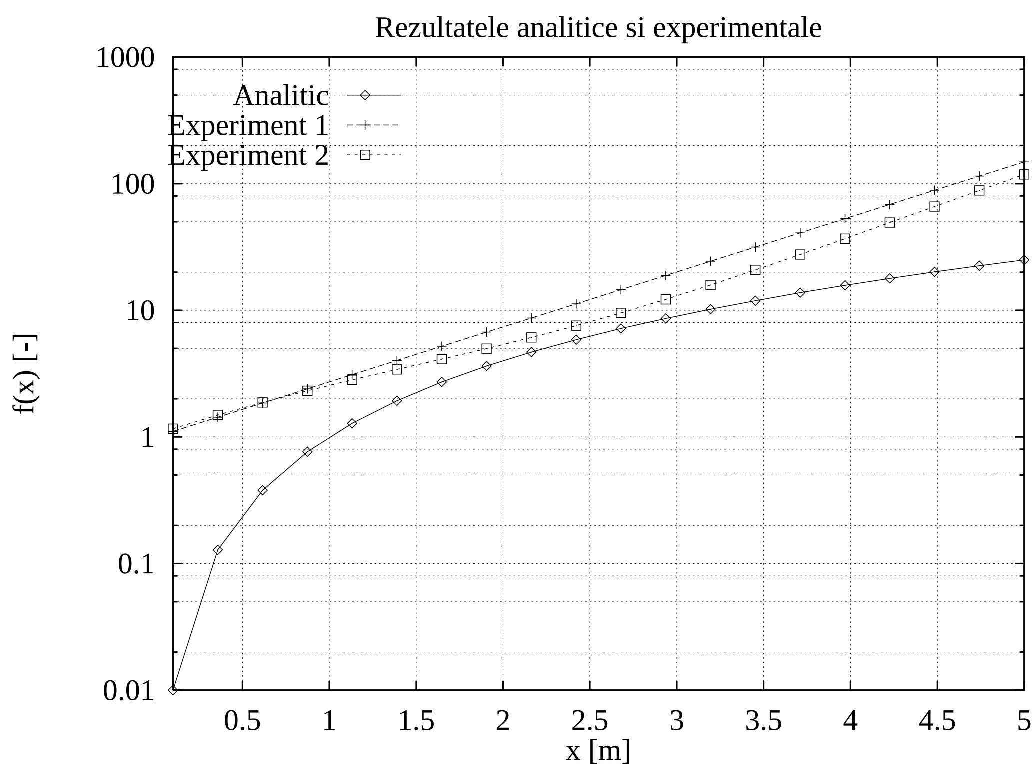


Fig. 5.3: Grafic realizat cu `gnuplot`

5.3 XPaint

5.3.1 Apelare

- În laboratorul EB212: `xpaint <↵ >`
- În laboratorul EB213: cu butonul din stânga al mouse-ului - Graphics - Xpaint sau cu comanda: `/comun/bin/GetXpaint <↵ >`

5.3.2 Ferestre xpaint

XPaint lucrează cu două ferestre:

- Fereastra de instrumente (toolbox window)
- Fereastra de desenare (painting window)

Fereastra de instrumente permite operaţii de intrare din fişier, deschiderea uneia sau mai multor ferestre de desenare, precum şi selectarea instrumentelor de desenare.

Fereastra de desenare permite realizarea desenului propriu-zis, precum şi operaţii de editare şi salvare în fişier a desenului.

5.3.3 Fereastra de instrumente

Aspectul acestei ferestre este prezentat în figura 5.4.

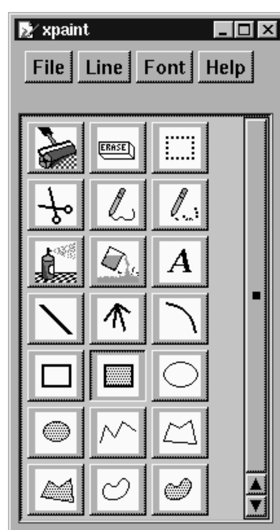


Fig. 5.4: Fereastra de instrumente (toolbox window) a programului XPaint

Fereastra conţine două zone:

- Zona de meniuri:
  - `File` – cu opţiunile de meniu: **New canvas** (nou desen cu dimensiuni predefinite); **New with size** (nou desen cu dimensiuni definite de utilizator); **Open**, **Load Clipboard** (deschiderea unui fişier existent); **Quit** (ieşire din XPaint);
  - `Line` – permite selectarea grosimii liniei de desenare;
  - `Font` – permite selectarea fontului şi dimensiunii caracterelor
  - `Help` – deschide o fereastră cu informaţii despre XPaint sau despre modul de utilizare al acestuia.
- Zona de butoane – prin care se selectează instrumentul de desenare. Printre aceste instrumente se numără: desenare punct cu punct, ştergere, desen forme geometrice regulate sau neregulate, colorare, etc.

5.3.4 Fereastra de desenare

Aspectul acestei ferestre este prezentat în figura 5.5.

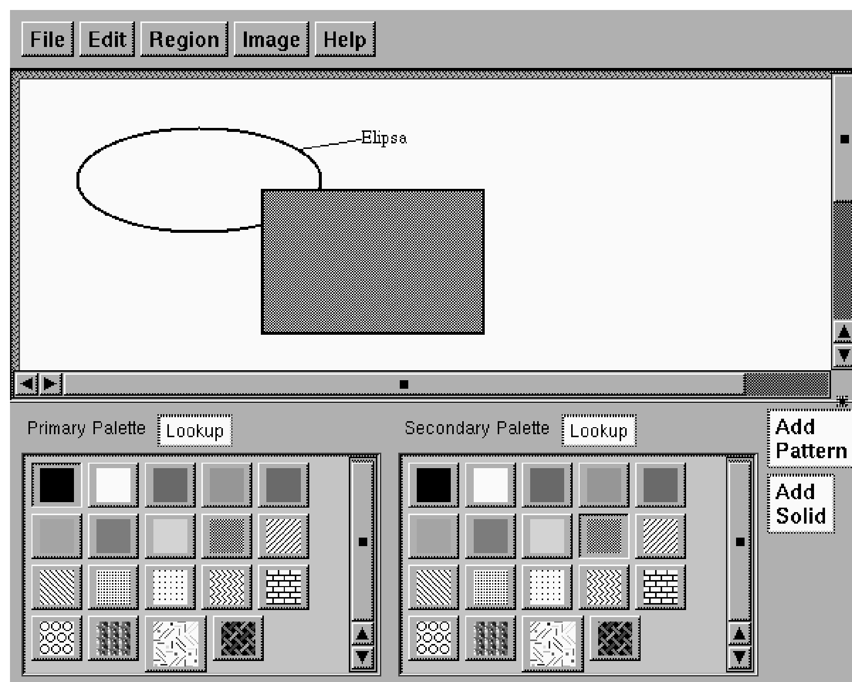


Fig. 5.5: Fereastra de desenare (painting window) a programului XPaint

Fereastra conţine trei zone:

- Zona de meniuri: Opţiunile din aceste meniuri permit efectuarea unor operaţii asupra desenului din fereastra respectivă, precum şi salvarea desenului în diferite formate.

- **File** – cu opţiuni pentru: salvare, salvare sub un alt nume, salvarea unei regiuni din desen, salvarea paletei de culori, încărcarea unei palete de culori creată anterior, ieşire din XPaint;
- **Edit** – cu opţiuni pentru: anularea ultimei comenzi, ştergerea unei zone şi memorarea ei în clipboard, copierea în clipboard a unei zone, copierea în desen a conţinutului clipboard–ului;
- **Region** – cu opţiuni pentru: reflectarea şi rotirea imaginii, aplicarea unor algoritmi standard de prelucrare a imaginilor (inversarea culorilor, estomparea sau accentuarea contururilor, crearea efectelor de relief sau de pictură în ulei);
- **Image** – cu opţiuni pentru: mărirea unei zone (lupă), setarea fineţei caroiajului imaginar pe care este plasată imaginea, schimbarea culorii fondului, modificarea dimensiunii imaginii;
- **Help** – afişarea instrucţiunilor de utilizare.

• Zona de desenare propriu–zisă;

• Zona de selectare a culorii liniilor şi respectiv a culorii de umplere a figurilor geometrice.

Atenţie: versiunea XPaint instalată în LMN mai are, se pare, unele bug–uri, care se manifestă prin ieşirea bruscă din program atunci când se comandă anumite operaţii. Vă recomandăm să salvaţi desenul mai des decât aţi face-o atunci când utilizaţi alte programe.

5.3.5 Salvarea desenului în diferite formate

XPaint oferă posibilitatea salvării desenului în 7 formate: TIFF, PPM, GIF, XBM, XPM, XWD şi PostScript (primele 6 formate sunt recunoscute şi ca formate de intrare).

Pentru a salva desenul în unul din aceste formate, se foloseşte meniul **File** al ferestrei de desenare şi opţiunea **Save as** a acestui meniu, cu selectarea formatului dorit prin apăsarea butonului corespunzător (figura 5.6).

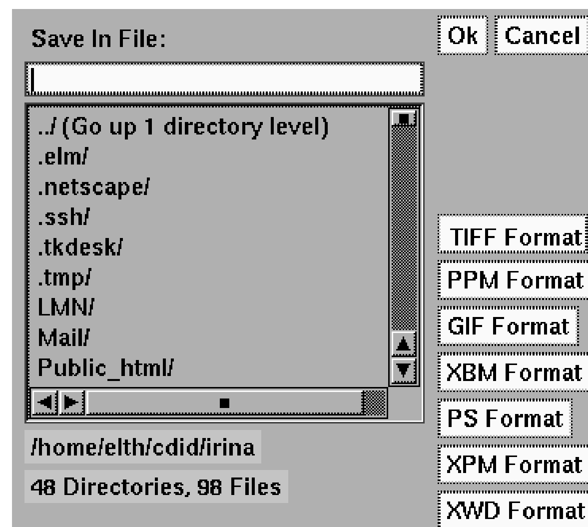


Fig. 5.6: Fereastra `xpaint` de salvare într-un fişier a desenului

5.4 XV

`xv` este un program interactiv care permite vizualizarea, modificarea şi conversia în alt format a imaginilor.

Printre formatele de intrare şi de ieşire se numără GIF, JPEG, TIFF, BMP, etc. Programul `xv` poate de asemenea genera fişiere PostScript.

Atenţie! Spre deosebire de celelalte programe prezentate până acum, `xv` nu este distribuit gratuit (*freeware*), ci este un produs promoţional (*shareware*), iar pentru folosirea sa profesională trebuie să cumpăraţi licenţa de utilizare. Pentru mai multe informaţii, consultaţi paginile de Help ale programului.

5.4.1 Apelare

- Cu butonul din stânga al mouse-ului – Graphics – xv
- Cu comanda

  `xv <↵ >`

5.4.2 Ferestre `xv`

Programul `xv` lucrează cu două ferestre:

- Fereastra de imagine – este cea care apare la lansarea `xv`;
- Fereastra de control – poate fi deschisă sau închisă prin click cu butonul din dreapta al mouse-ului în fereastra de imagine.

În fereastra de imagine se află imaginea în curs de vizualizare/modificare. Comenzile de modificare a imaginii sunt disponibile în fereastra de control.

5.4.3 Operaţiuni în fereastra de imagine

În această fereastră `xv` sunt posibile trei operaţii:

- **Apăsând butonul din mijloc al mouse-ului** – Afişarea informaţiilor referitoare la imagine: poziţia cursorului în pixeli, culorile şi saturaţia pixel-ului curent;
- **Apăsând butonul din dreapta al mouse-ului** – deschiderea / închiderea ferestrei de control;
- **Apăsând butonul din stânga şi deplasând mouse-ul** – Selectarea unei porţiuni din imagine, asupra căreia vor acţiona prelucrările ulterioare.

5.4.4 Operaţiuni în fereastra de control

Fereastra de control **xv** are aspectul prezentat în figura 5.7.

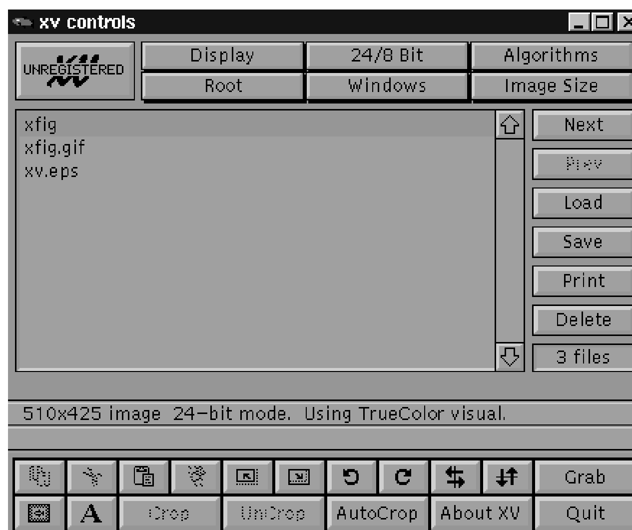


Fig. 5.7: Fereastra de control a programului **xv**

Fereastra de control cuprinde o zonă în care este afişată **lista fişierelor**–imagine deschise sau salvate în sesiunea curentă, precum şi **trei zone de butoane şi meniuri**.

Zona superioară conţine **meniuri** pentru efectuarea unor operaţii asupra imaginii:

- **Display** – permite schimbarea aparenţei imaginii pe ecran, fără modificarea acesteia; meniul conţine opţiuni precum Raw (forma iniţială); Smooth (forma "netezită", compensând erorile de rotunjire care intervin la mărirea/micşorarea imaginii); Dithered (culori cât mai apropiate de cele originale); etc.
- **Root** – permite schimbarea fondului întregului ecran
- **24/8 Bit** – trece imaginea din formatul cu 24 biţi/pixel în cel cu 8 biţi/pixel şi invers.
- **Windows** – deschide şi alte ferestre, precum: Visual Schnauzer (un manager de fişiere–imagine); Color Editor (editor de culori); Text View (vizualizarea conţinutului fişierului imagine, în format ASCII); etc.
- **Algorithms** – permite aplicarea unor algoritmi standard de prelucrare a imaginilor, precum: Blur (estomparea contururilor), Sharpen (accentuarea contururilor), Oil Painting (efect de pictură în ulei), Copy/Clear Rotate (rotirea imaginii), etc.
- **Image size** – permite modificarea dimensiunii imaginii.

Zona din dreapta este dedicată **lucrului cu fişiere** şi conţine butoanele:

- **Next** – deschide următorul fişier din lista de fişiere ale sesiunii curente;
- **Prev** – deschide precedentul fişier din lista de fişiere ale sesiunii curente;
- **Load** – încarcă un nou fişier de pe disc;
- **Save** – salvează fişierul curent în formatul original sau în alt format;
- **Print** – imprimă un fişier imagine;
- **Delete** – şterge un fişier, fie din lista de fişiere ale sesiunii, fie de pe disc.

Zona inferioară a ferestrei de control conţine **butoane** pentru modificarea imaginii. O parte dintre butoane sunt active numai dacă a fost selectată o zonă din imagine. Butoanele din zona inferioară au următoarele funcţii:

- **Rândul de sus**:
  1. memorarea în clipboard a unei copii a zonei selectate;
  2. ştergerea zonei selectate (colorarea ei în culoarea fondului) şi copierea în clipboard a zonei şterse;
  3. copierea ("Paste") a imaginii din clipboard; butonul trebuie apăsat de două ori: prima dată pentru a selecta zona în care să se facă copierea, şi a doua oară pentru a realiza copierea propriu-zisă;
  4. ştergerea zonei selectate;
  5. micşorarea întregii imagini cu 10%;
  6. mărirea întregii imagini cu 10%;
  7. rotirea la stânga cu 90 grade;
  8. rotirea la dreapta cu 90 grade;
  9. reflectarea imaginii faţă de axa verticală;
  10. reflectarea imaginii faţă de axa orizontală;
  11. "capturarea" unei imagini de pe ecran şi afişarea ei în fereastra de imagine, cu variantele: Grab (capturează o fereastră cu butonul din stânga, o zonă dreptunghiulară cu butonul din mijloc); AutoGrab (după un interval prestabilit, capturează fereastra în care se află cursorul mouse-ului)
- **Rândul de jos**
  1. adaugă o margine imaginii curente, cu dimensiuni şi culoare selectabile de utilizator;
  2. adaugă un text;
  3. elimină marginea din jurul zonei selectate (Crop);
  4. anulează efectul comenzilor Crop şi AutoCrop;

5. elimină marginea din jurul zonei de interes a imaginii (AutoCrop);
6. afişează un text despre `xv`;
7. iese din `xv`.

5.5 Exerciţii

EXERCIŢIUL 1

Folosind `gnuplot`, realizaţi graficul funcţiei $\sin x/x$, pe intervalul [-15, 15]. Adăugaţi titlu şi etichete pentru axe.

EXERCIŢIUL 2

Realizaţi cu `gnuplot` graficul funcţiei $f : [-10, 10] \times [-1, 3] \to$ R, $f(x, y) = sin(x) * cos(y)$, reprezentat în figura 5.8. Adăugaţi toate elementele care apar în figura 5.8 (titlu, etichete, săgeată, etichete axe).

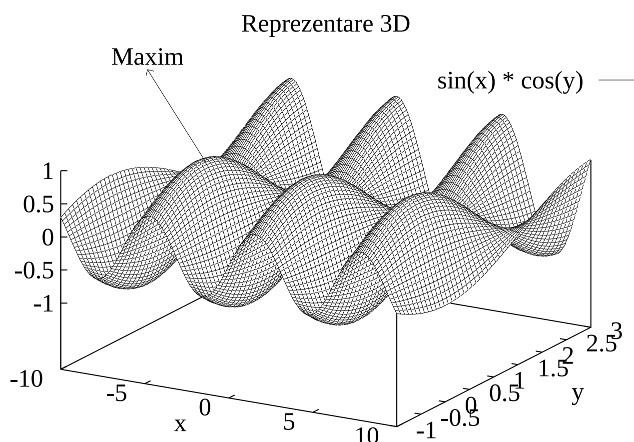


Fig. 5.8: Graficul funcţiei $sin(x) * cos(y)$

Indicaţie: folosiţi parametrii `isosamples, xlabel, title, arrow, label`.

EXERCIŢIUL 3

Completaţi paragraful 2 al referatului de laborator realizat la exerciţiul 6 din capitolul 4 cu încă o ecuaţie: formula intensităţii curentului prin circuitul RC serie.

Folosind `gnuplot` reprezentaţi pe acelaşi grafic tensiunea la bornele condensatorului şi intensitatea curentului, pentru valori arbitrare ale parametrilor R, C şi E şi pentru un interval de timp convenabil ales.

Adăugaţi la paragraful 2 al referatului o figură cu acest grafic, precizând într-o frază valorile parametrilor şi făcând o referire la figură. (Ca orice figură dintr-un text ştiinţific, şi aceasta va fi numerotată şi va avea o scurtă explicaţie.)

EXERCIŢIUL 4

Realizaţi cu XFig schema circuitului RC serie descris în referatul de laborator (fişierul `rc.tex`).

Includeţi schema în paragraful 2, având grijă ca figura să aibă un număr, un titlu, şi să fie citată în textul referatului.

EXERCIŢIUL 5

Cu ajutorul programului `xv`, realizaţi o aşa–numită "captură de ecran" a ferestrei `xfig` în care aţi realizat exerciţiul precedent.

Opţional: Editaţi această imagine cu `xpaint`, indicând prin săgeţi şi nume câteva din regiunile ecranului XFig prezentate în paragraful 5.1.3.

Încercaţi diferitele comenzi `xv` (algoritmi, culori, modificare dimensiuni, etc.) şi observaţi efectul acestora.

Când rezultatul este satisfăcător, salvaţi imaginea în format PostScript, încercând să obţineţi un fişier cât mai mic dar şi o bună calitate a imaginii (încercaţi diferite variante: culori, dimensiuni ale imaginii). Atenţie, dacă imaginea nu se vede bine pe ecran, nu înseamnă că acelaşi lucru se va întâmpla şi pe hârtie!

Adăugaţi o primă anexă la referatul Dvs: **Programul folosit pentru desenarea schemei electrice**, care va conţine un scurt comentariu urmat de figura ecranului XFig.

Capitolul 6

Analiza numerică a modelelor matematice — **Scilab** —

Specific activităţii de cercetare ştiinţifică este faptul că se urmăreşte caracterizarea cantitativă a sistemelor şi fenomenelor studiate. Pentru a realiza acest deziderat, în inginerie se foloseşte modelarea abstractă, bazată pe următoarele etape:

- **modelarea fizică**, în care se identifică fenomenele esenţiale, ipotezele simplificatoare şi se stabilesc mărimile fizice caracteristice studiului respectiv;
- **modelarea matematică**, în care relaţiile stabilite anterior sunt aduse într-o formă matematică şi se formulează corect o problemă care, în esenţă, presupune rezolvarea unui sistem de ecuaţii algebrice liniare sau neliniare, diferenţiale, integrale sau cu derivate parţiale;
- **modelarea numerică**, urmăreşte rezolvarea problemei matematice formulate anterior, până la obţinerea rezultatelor numerice susceptibile de a fi comparate cu cele experimentale sau determinate pe altă cale.

Deoarece problemele actuale au o complexitate tot mai mare, pentru a putea obţine rezultate de nivel înalt în cercetarea ştiinţifică şi dezvoltarea tehnologică este absolut necesară utilizarea tehnicii moderne de calcul în studiul situaţiilor reale, întâlnite în practică. Din acest motiv s-a conturat un nou domeniu al ştiinţei intitulat "*Computational Science and Engineering – CSE*", care complementează abordările teoretice şi cele experimentale.

Încă de acum două decenii, cercetătorii s-au confruntat cu dificultatea lucrului în echipe interdisciplinare (care conţin matematicieni şi informaticieni) sau cu necesitatea căpătării unor cunoştinţe profunde de metode numerice şi de programare a calculatoarelor. În acest context au fost create instrumente software specializate (cum sunt *Mathematica*

[81] şi *MATLAB* [51]), care permit cercetătorului să se concentreze cu mare productivitate asupra problemei supusă studiului şi care ascund detaliile matematice şi de programare. Aceste programe sunt sisteme software complexe (pachete polifuncţionale integrate) bazate pe un interpretor (asemănător BASIC-ului), care recunoaşte comenzi ce permit:

- **calcule numerice**, evaluări de expresii matematice;
- **vizualizarea** datelor, reprezentări grafice de funcţii;
- realizarea şi execuţia unor **programe de calcul** într-un limbaj specializat, de nivel înalt;
- o **interfaţare** grafică prietenoasă cu utilizatorul, interfaţarea cu alte programe (controlul programelor şi proceselor externe, controlul din exterior al proceselor interne, importul şi exportul de date care să conducă la realizarea de documente profesionale PostScript sau TEX).

Unele pachete oferă şi funcţia de calcul simbolic (implicit în Mathematica şi opţional în MATLAB).

Utilizarea unor astfel de pachete specializate sporeşte incomparabil productivitatea cercetătorului, deoarece pe de o parte algoritmii matematici folosiţi de ele sunt cei mai performanţi, iar pe de altă parte, limbajul de programare este simplu (nu necesită declaraţii de date).

De exemplu, în *MATLAB*:

- obiectul fundamental este matricea de numere complexe, care, dacă are o singură coloană devine vector, iar dacă are un singur element devine scalar, urmând ca prin anularea părţii imaginare, el să devină număr real;
- dimensiunile matricelor nu sunt declarate explicit, ci alocarea memoriei se efectuează dinamic pe măsura necesităţilor, iar operaţia de eliberare a memoriei ("*garbage collection*") este tot automată;
- limbajul de programare *MATLAB* oferă toate structurile de control specifice unui limbaj de programare modern;
- expresiile admise de limbaj au mult mai multă putere decât la un limbaj universal: operanzii sunt matrice, iar operatorii pot realiza operaţia de înmulţire, inversare (sau de pseudoinversă) sau funcţii de matrice;
- operaţiile simple (calcule aritmetice, reprezentări grafice) se realizează practic instantaneu şi cu un efort minim de tastare din partea utilizatorului.

În laboratorul LMN, pe lângă *Mathematica* şi *MATLAB*, este instalat un astfel de mediu numit **Scilab** (se citeşte *silab*), realizat şi distribuit gratis de INRIA – *Institut National de Recherche en Informatique et Automatique* [22]. Acest mediu emulează şi extinde funcţiile *MATLAB* permiţând şi manipularea matricelor ale căror elemente sunt:

- şiruri de caractere;
- polinoame cu coeficienţi eventual complecşi;
- funcţii raţionale (raport de polinoame);
- liste cu obiecte de tipuri diferite,

ceea ce permite, în final, manipularea simbolică a sistemelor liniare în sensul teoriei sistemelor. Programul permite interfaţarea cu limbaje ca FORTRAN şi C sau cu programe precum MAPLE V [56] pentru realizarea calculelor simbolice şi conţine şi interfeţe pentru descrierea grafică şi simularea sistemelor dinamice (*SCICOS* — mediu asemănător *SIMULINK*-ului).

Obiectivele capitolului sunt:

- familiarizarea cu funcţiile şi conceptele unui mediu de înaltă productivitate de tip "laborator ştiinţific" sau "laborator matematic";
- însuşirea deprinderilor necesare analizei numerice a unui sistem simplu (de exemplu un circuit electric) şi redactarea unui raport profesional folosind funcţiile "laboratorului ştiinţific".

6.1 Apelare Scilab

- În laboratorul EB212: `scilab <↵ >`
- În laboratorul EB213:

  butonul din mijloc - Mathematics - Scilab

  sau

  `/comun/bin/GetScilab`

În LMN este instalat pe maşinile lmnhp1, lmnhp4 şi pe toate maşinile Linux; necesită mediul X11 pentru a putea fi executat.

6.2 Componente Scilab

- Interpretor – permite introducerea în mod interactiv a unor comenzi de la consolă şi executarea imediată a acestora
- Biblioteci de macrouri - proceduri Scilab
- Biblioteci de rutine C şi FORTRAN - externe programului Scilab, apelabile de acesta

6.3 Interfaţa grafică a programului Scilab

În figura 6.1 este prezentată interfaţa grafică a programului Scilab.

Pe lângă zona de comandă, în care utilizatorul poate introduce de la consolă comenzi şi instrucţiuni Scilab, interfaţa mai conţine butoane pentru realizarea următoarelor funcţii:

- Operaţii cu fişiere şi directoare (schimbarea directorului curent, încărcarea unui fişier conţinând un program Scilab, încărcarea unui fişier conţinând definiţiile unor funcţii Scilab, oprirea execuţiei programului Scilab, etc.)
- Operaţii de control al execuţiei (oprirea temporară sau definitivă a comenzii în curs de execuţie, continuarea execuţiei comenzii)
- Lansarea programelor demonstrative
- Operaţii cu ferestrele grafice (creare, închidere, definirea ferestrei curente)
- Lansarea ferestrei de Help

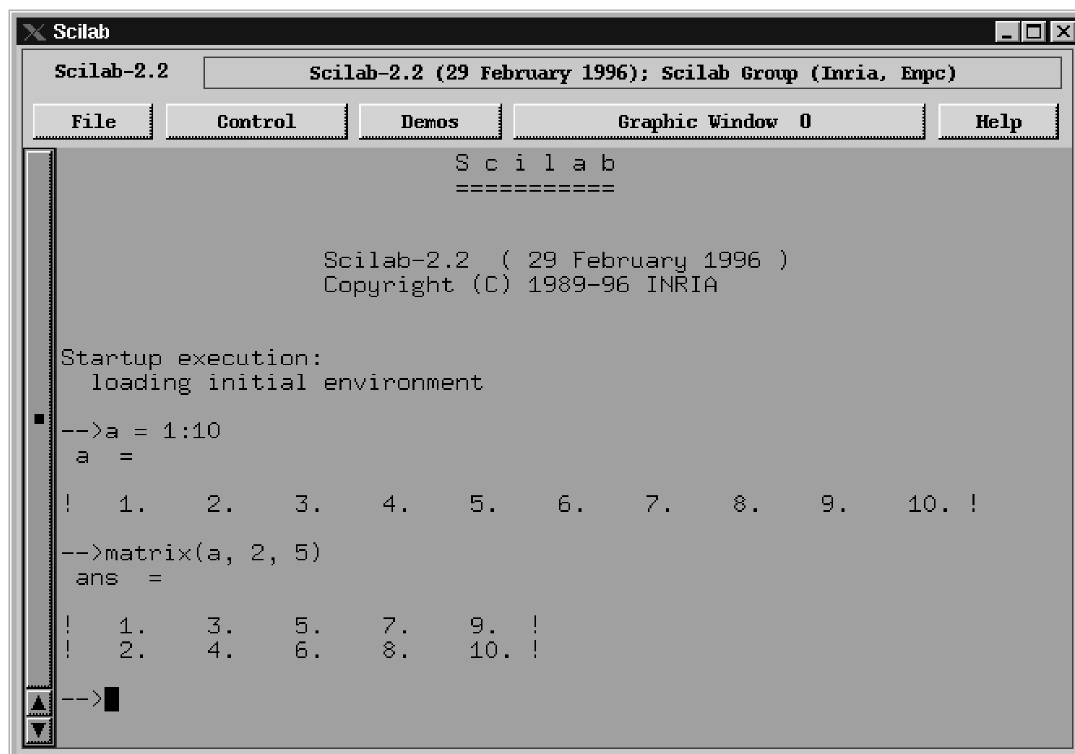


Fig. 6.1: Interfaţa grafică a programului Scilab

6.4 Câteva noţiuni introductive

6.4.1 Formatul comenzilor Scilab

- Comentarii: tot ceea ce urmează pe o linie după caracterele //

- În mod normal, după ce se dă o comandă, se afişează rezultatul imediat după apăsarea tastei `<↵ >`. Pentru a se executa comanda fără afişarea rezultatului, se pune ; (punct şi virgulă) la sfârşitul instrucţiunii.

- Se pot da mai multe comenzi pe o linie, separate prin ;

- Dacă nu se precizează cărei variabile să i se atribuie rezultatul, printr-o instrucţiune de forma:

  ```
  var = expresie
  ```

  ci se invocă doar `expresie`, atunci valoarea acesteia este memorată în variabila implicită `ans`.

- În fişierul de configurare `~/.scilab` se pot scrie comenzi care să se execute de fiecare dată la pornirea Scilab (de exemplu comenzi de tip `getf`, vezi paragraful 6.6.1).

6.4.2 Facilităţi

- manipulare de matrice cu elemente reale sau complexe de tip:
  - scalari
  - vectori / matrice
  - polinoame
  - şiruri de caractere
  - liste
- calcule simbolice cu polinoame şi cu matrice de polinoame
- calcule simbolice cu sisteme liniare
- simulări, optimizări
- interfaţă cu programe scrise în C şi în FORTRAN.

Tipurile de date sunt, aşadar, variate, flexibile. Sintaxa comenzilor este naturală şi uşor de utilizat. Există un număr mare de primitive standard şi se pot adăuga altele noi de către utilizator.

6.4.3 Demonstraţie

Selectând cu mouse-ul butonul "Demos" din meniu se lansează un set de programe demonstrative pe care vă recomandăm să le executaţi pentru a înţelege rapid filozofia Scilab. Sunt vizualizate instrucţiunile Scilab utilizate în demonstraţie precum şi rezultatele lor.

6.5 Tipuri de date Scilab

6.5.1 Constante

Există constante predefinite. Numele lor incepe cu `%`:

- `%i` $= \sqrt{-1}$;
- `%pi` $= \pi$;
- `%e` $= e = 2.7182818...$;
- `%eps` – precizia masinii, numărul care satisface egalitatea numerică `1 + %eps = 1`;
- `%inf` – infinit;

- `%nan` = "not a number"

Constante de tip boolean:

- `%t` – true (rezultatul comparaţiei 1 == 1);
- `%f` – false.

6.5.2 Matrice de scalari

Tipul "matrice" este cel mai important tip de date în Scilab. Orice obiect de tip: scalar, vector, matrice în sens clasic, face parte din clasa "matrice" iar operaţiile cu ele se efectuează în mod unitar, ca operaţii cu matrice. În particular, un scalar este o matrice de dimensiune 1×1.

1. **Scalari**

   Ex:
   ```
   a = 2
   b = 1 + 3* %i     // 1 + 3j
   B = -1 + 3* %i    // B şi b sunt două variabile diferite
   ```
   Se pot aduna, scădea, împărţi, ridica la putere (cu operatorul ^, de exemplu: `x^2`).

2. **Vectori**

   - Incluşi între paranteze pătrate,
     - cu blanc sau virgulă între elemente: vectori linie

       Ex: `v = [1 2 1+%i]` sau

       `v1 = [1, 2, 1+%i]`
     - cu ; între elemente: vectori coloană
   - Operaţii:

`+ - * /`	- în măsura în care dimensiunile vectorilor sunt compatibile
`'`	- transpunere, elementele complexe conjugate prin transpunere
`v1 / v2`	- echivalent cu `v1 * pinv(v2)`, unde `pinv(v2)` este pseudoinversa (`v2*pinv(v2) = 1`)
`.*`	- înmulţire element cu element
`./`	- împărţire element cu element

   - Crearea automată a unui vector:

     `v = val_in : pas : val_fin`

     `pas` poate lipsi; în acest caz el este considerat implicit egal cu 1.

3. **Matrice**

- Definite astfel: elementele de pe o linie sunt separate prin blanc sau prin virgulă, iar între linii se pune caracterul ;

  Ex: `a = [2 3; 4 5]` // Prima linie: 2, 3; A doua linie: 4, 5
- Operaţii disponibile:

  `+ - *` - operaţii matriceale, dimensiunile operanzilor trebuie să fie compatibile

  `.* ./` - înmulţire, respectiv împărţire element cu element (rezultă o matrice de aceeaşi dimensiune)

  `b / A` - are semnificaţia: soluţia sistemului de ecuaţii Ax = b
- Se pot forma matrice din vectori sau din alte matrice (concatenare). Ex:
  ```
  a = 1:10
  b = [a, a, a]
  b1 = [a; a; a]
  ```
- Se pot reorganiza elementele existente într-o matrice astfel încât să se modifice numărul de linii / coloane.

  Ex: `c = matrix (b, 3, 4)` // o nouă matrice cu 3 linii şi 4 coloane
- Transpunere: `a'`
- Determinant: `det(a)`

6.5.3 Matrice de şiruri de caractere

- Şir de caractere: inclus între apostroafe (′)
- Operaţii cu şiruri:

  `string(nr)` - valoarea numerică `nr` scrisă ca şir

  `addf(sir1, sir2)` - crează şirul sir1+sir2

  `sir1 + sir2` - concatenează şirurile `sir1` şi `sir2`

  ..., etc.
- Matricele de şiruri de caractere permit:
  - crearea şi manipularea funcţiilor utilizator
  - manipularea simbolică a unor obiecte matematice

Exemplu

```
a = 2; b = 1; c = 3; d = 5
b1 = 1; b2 = 2;

A = ['a' 'b'; 'c' 'd']
```

```
B = ['b1'; 'b2']
evstr(A)                  // evalueaza 'in a, b, c, d
                          // matricea de siruri  A
At = trianfml(A)          // triangularizeaza matricea A
                                 // -- formula, nu numeric
[x] = trisolve(At, B)     // rezolva simbolic
                                 // sistemul  At x = B
```

6.5.4 Matrice de polinoame şi de funcţii raţionale

- Polinoamele se definesc:
  - specificând rădăcinile:
    `polinom = poly([listă de rădăcini], 'var', 'roots')`,
    unde `'roots'` poate lipsi, fiind implicit.
    Ex: `p1 = poly([1 2], 's')`
  - specificând coeficienţii: `polinom = poly([listă de coeficienţi], 'var', 'coeff')`.
  - în forma clasică, după definirea anterioară a variabilei polinomului, ca fiind un alt polinom cu o singură rădăcină zero:
    `z = poly(0, 'z')`
    `p2 = 1 + 2*z + 5*z^2 + ...`
- Funcţie raţională: raportul a două polinoame

  Ex: `r = p1/p2`
- Se pot construi matrice din polinoame şi din funcţii raţionale.

  Ex: `a = [1, p; p+3, p^2]`
  `b = [1, p; 1/p, (p+1)/p]`

Se recomandă să se folosească forma cu `,` (virgulă) pentru a separa elementele de pe o linie.

6.5.5 Matrice cu elemente de tip logic

Conţin elemente de tipul `%t`, `%f` sau variabile logice. Asupra lor se pot efectua operaţiile obişnuite cu matrice: concatenare, transpunere, operaţiile `or` şi `and`.

6.5.6 Liste

Matricele sunt colecţii de obiecte de acelaşi tip, pe când **listele** sunt colecţii de obiecte de tipuri diferite. Listele sunt folosite pentru a defini date structurate (structuri sau înregistrări).

Printre listele predefinite un rol important îl joacă sistemele liniare în sensul teoriei sistemelor.

- Forma generală:

  `l = list(a1, a2, ..., an)`

  Fiecare `ai` este un obiect Scilab (matrice, listă, ..)

- Operaţii cu liste:

inserare:	`l(i) = a`
ştergere:	`l(i) = null()`
extragerea unor obiecte:	`[o1, o2, ...] = l(v)`, unde: `v` este un vector de indici iar `o1, o2`, ... sunt elementele din listă, corespunzătoare indicilor.

6.6 Programarea în Scilab

6.6.1 Funcţii Scilab

Funcţiile Scilab (numite şi macrouri) pot fi definite într-un fişier sau on-line.

- În fişier:

```
function [y1, y2, ..., yn] = nume_fct(x1, x2, ..., xm)
                  // y1 .. = rezultate; x1 .. = parametri
         instrucţiuni
end
```

- On-line:

```
deff ('[y1, ..., yn] = nume_fct(x1 , ..., xm)', ...
    'instrucţiuni')
```

- Citirea unui fişier conţinând macrouri: `getf('nume_fis')`

6.6.2 Funcţii predefinite în Scilab

Printre funcţiile predefinite se numără:

- Funcţiile matematice clasice (`sin, cos, log, ...`)
- Rutine specializate pentru rezolvarea unor probleme matematice, de tipul:
  - ecuaţii diferenţiale (`ode, impl, dassl, ...`)
  - optimizare (`optim`)
  - funcţii de algebră liniară (`svd, qr, det, lu, schur, ...`)
  - integrare numerică (`integrate`)
  - ... şi multe altele. Puteţi obţine lista completă a comenzilor folosind **Help**.
- Funcţii pentru supraîncărcarea (*overloading*) unor operatori (precum `+, -, *`), care trebuie să acţioneze în mod diferit asupra unor tipuri diferite de operanzi (realizându-se astfel polimorfismul acestor operatori). Numele lor începe cu `%`.

Fişierele care conţin funcţii predefinite pot fi atât în format cod–sursă (cu extensia `.sci`), fie compilate, în format binarA (cu extensia `.bin`).

6.6.3 Tipuri de instrucţiuni Scilab

Sunt disponibile (lista nu e completă însă) următoarele tipuri de instrucţiuni:

- Expresii cu operatori matematici: `+ - * / \`
- Expresii cu operatori de relaţie (pentru comparaţii):

  `==` sau `=` pentru testarea egalităţii,

  <, >, <=, >= pentru testarea ordinii,

  <> sau `~=` pentru testarea inegalităţii
- Operaţii de intrare/ieşire (`print, printf, read, scanf`)
- Decizii:
  - simple:

    `if` *condiţie* `then, ..., else, ..., end`
  - multiple:

```
select nume_var,
    case val1, instrucţiuni,
    case val2, instrucţiuni,
```

```
      . . .
      else, instrucţiuni
    end
```

- Cicluri
  - cu contor:
    ```
    for contor = vmin:vmax, instrucţiuni, end
    ```
    sau:
    ```
    v = [lista elemente]
    for contor = v, instrucţiuni, end
    ```
  - cu test iniţial:
    ```
    while condiţie, instrucţiuni, end
    ```
- Instrucţiuni de oprire a ciclurilor

`pause`	- se intră în alt nivel (alt bloc)
`return, return(val)`	- se revine la nivelul anterior
`resume`	- se revine la nivelul anterior
`abort`	- opreşte execuţia functiei
`setbpt`	- setare breakpoint
`delbpt`	- ştergere breakpoint
`disbpt`	- afişare breakpoint

- Apeluri de funcţii, predefinite sau definite de utilizator (vezi paragrafele anterioare) – se realizează prin invocarea numelui funcţiei urmat de parametrii actuali.

6.7 Fişiere de comenzi

Fişierele de comenzi se pot scrie off-line cu un editor de text, sau prin salvarea definiţiilor sesiunii curente (variabile, funcţii, etc.). Se recomandă folosirea extensiei implicite `.sci`.

- Afişarea variabilelor definite: `who`
- Lista de comenzi şi funcţii sistem definite: `what`
- Help: click pe "Help" din meniu sau cu comanda `help` *cuvânt_cheie*
- Salvarea variabilelor în fişier extern (binar): `save`
- Citirea unui fişier de variabile: `load`
- Scrierea unei variabile (obiect Scilab) într-un fişier, în format FORTRAN:

```
write ( 'nume_fis', var)
```

- Citirea :

  `var = read ( 'nume_fis', ` *format*) unde

  "format" e de tipul: nr. linii, nr. coloane

6.8 Grafice

Graficele se pot afişa în ferestre X şi se pot salva pe disc în trei formate, corespunzătoare driverelor: `X11`, `Pos` (PostScript) şi `Xfig`. Mai există şi driverul `Rec`, care afişeaza graficul în fereastra X şi îl şi salvează în fisier.

- Selectarea unui tip de driver:

  `driver ('nume_driver')`

- Comanda `plot..(...)` afişează graficul în fereastra 0; dacă driverul selectat este `rec`, scrie şi instrucţiuni PostScript în fişierul `foo.ps`

- Pentru a selecta un fişier PostScript, altul decât `foo.ps` se foloseşte comanda:

  `xinit('fis.ps')`

- Comenzi de desenare de grafice 2D:

  `plot`: grafic a 2 vectori, y ca funcţie de x

  `plot(x, y, <et_x, et_y, titlu>)`

  `plot2d` - plot multiplu: parametrii sunt matrice de aceeaşi dimensiune; se iau coloanele pe rând şi se face câte un grafic.

  `plot2d(x, y, <stil linie>, "pqr", "etx@ety@titlu" ...)`

  p = 0, 1 (afişare sau nu etichete)

  q = 0 .. 4 (scalare axe)

  r = 1, 2 (frontieră grafic – border)

  Suprapunerea mai multor curbe pe acelaşi grafic se face dând pe rând comenzile de realizare a fiecărui grafic, fără ştergerea ferestrei grafice:

  parametrul al patrulea: "p0r" (nu şterge fereastra grafică)

  `plot2d1` - permite şi axe logaritmice

  `plot2d2` - funcţii constante pe porţiuni

  `plot2d3` - grafic cu bare verticale

  . . .

- Comenzi de desenare de grafice 3D:

`plot3d` - grafice tridimensionale

Ex: `plot3d(x, y, z, teta, alfa, "etx@ety@etz", <...>)`

`plot3d1` - nivele de gri

`plot3di` - modificarea interactivă a poziţiei de vizualizare cu butonul din stânga, oprire cu butonul din dreapta

- Se pot face grafice în mai multe ferestre sau se pot trimite în fişiere cu nume selectat de utilizator.
- Se pot afişa mai multe grafice independente în aceeasi fereastră cu comanda `xsetech`

Observaţii:

1) Salvarea unui grafic într-un fişier, în formatul dorit, se poate face şi direct din fereastra X în care este afişat graficul, cu butonul File → Export.

2) Dacă formatul de ieşire dorit este `EPS` şi vreţi să realizaţi mai multe grafice în aceeaşi sesiune Scilab, trebuie să introduceţi comanda

```
xbasc()
```

(ştergere fereastră grafică) după fiecare grafic, altfel, în fişier se vor salva toate comenzile de grafică introduse de la începutul sesiunii Scilab.

3) O variantă robustă şi care permite realizarea cu uşurinţă a unor grafice cu aspect profesional este salvarea graficelor în format XFig, urmată de editarea şi salvarea lor în format EPS, din XFig.

6.9 Exemple

În figura 6.2 este prezentat graficul din figura 5.3, realizat de astă dată cu Scilab. Instrucţiunile Scilab sunt următoarele:

```
x = [0.1:0.2:5]; x = [x, 5];

f = x^2;
y = read("grafic.dat", -1, 3);
xa = y(:, 1);
f1 = y(:, 2);
f2 = y(:, 3);

xs = [x' xa' xa'];
fx = [f' f1' f2'];

xset("use color", 0)
xset("dashes")
```

```
xset("font",2,3)
xbasc()
plot2d1(´gnl´, xs, fs, [1 2 3])      // cu puncte
plot2d1(´gnl´, xs, fs, [-1 -2 -3],´102´,´Analitic@Experiment1@Experiment2´)
                                   // cu linii si etichete grafice
xgrid()
xtitle("Rezultatele analitice si experimentale")
```

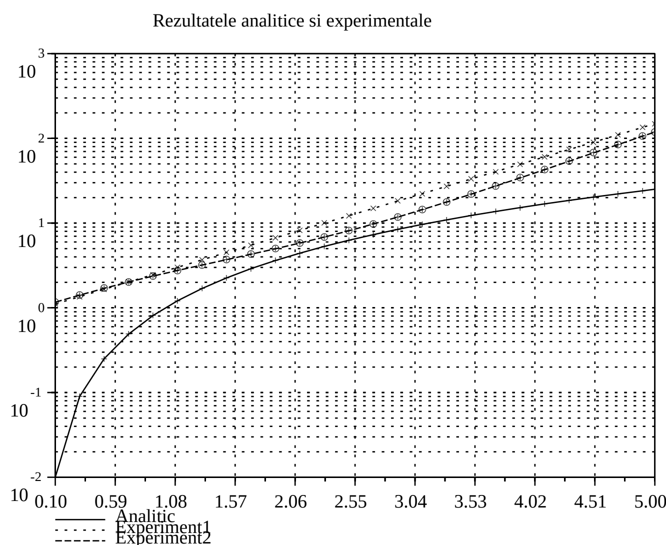


Fig. 6.2: Graficul din fig. 5.3, realizat cu Scilab

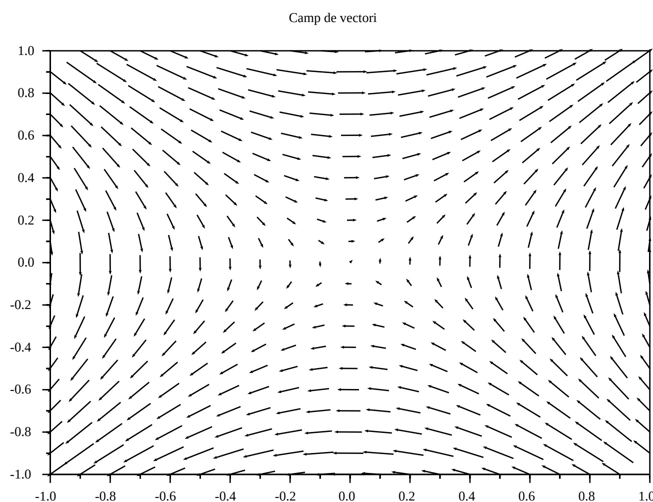


Fig. 6.3: Câmp de vectori reprezentat grafic cu Scilab

În figura 6.3 este prezentat un grafic realizat cu Scilab, folosind următoarele instrucţiuni:

```
x = [-1:0.1:1]; y=x; u = ones(x);
fx = x .*. u´;  fy = u .*. y´;
xbasc();
champ(x, y, fx, fy);
xtitle(´Camp de vectori´)
```

6.10 Apelul unei rutine C sau FORTRAN din Scilab

Deşi Scilab dispune de un număr mare de funcţii predefinite, acestea nu acoperă toată gama de probleme matematice de care un cercetător are nevoie în activitatea sa. Pentru a evita necesitatea de a “traduce” în limbajul Scilab o rutină de calitate deja existentă, dar scrisă în FORTRAN sau în C, Scilab oferă facilitatea de a apela rutine externe din bibliotecile matematice.

Pentru ca o rutină externă să poată fi apelată din Scilab, aceasta trebuie să îndeplinească câteva condiţii:

- Să fie deja compilată, în format obiect (cu alte cuvinte, să fie conţinută într-un fişier obiect, de obicei cu extensia `.o` sau `.a`).
- Să fie o subrutină (în cazul rutinelor FORTRAN), respectiv o funcţie de tip `void` (în cazul rutinelor C).

  În cazul funcţiilor FORTRAN, sau funcţiilor C care întorc o variabilă de orice tip cu excepţia `void`, apelul poate fi făcut din Scilab, dar nu se va putea accesa valoarea întoarsă de funcţie.
- În cazul funcţiilor C, toţi parametrii trebuie să fie transmişi prin adresă şi nu prin valoare.

Aceste condiţii sunt impuse de faptul că tipul implicit al rutinelor externe apelate din Scilab este “FORTRAN”, deci aceste funcţii trebuie să îndeplinească toate cerinţele compatibilităţii C–FORTRAN (vezi paragraful 7.5).

Pentru a **extinde lista de rutine din Scilab** cu o rutină externă şi pentru a apela o astfel de rutină trebuie parcurse următoarele **etape**:

- Se defineşte legătura (“link”) la rutina externă:

  ```
  ll = link('cale/bibl.o', 'rutina', 'F')
  ```

  în cazul rutinelor FORTRAN, respectiv:

  ```
  ll = link('cale/bibl.o', 'rutina', 'C')
  ```

  în cazul rutinelor C.

  În comenzile de mai sus, `cale` reprezintă o cale spre directorul în care se găseşte fişierul obiect `bibl.o`, iar `rutina` reprezintă numele rutinei.

Pentru a afişa lista tuturor rutinelor externe, se foloseşte aceeaşi comandă, fără parametri:

```
link()
```

Pentru a elimina o rutină din lista de rutine externe se foloseşte comanda (disponibilă în versiunile mai recente decât 2.2):

```
ulink(ll)
```

- Se apelează rutina cu o comandă de forma:

```
[y1,y2, ...]  = fort('rutina', intrari, 'out', iesiri)
```

în care:

  - `y1, y2, ...` reprezintă numele unor variabile Scilab în care vor fi memorate ieşirile rutinei
  - `fort` este numele comenzii Scilab de apel al unei rutine externe
  - `rutina` este numele rutinei externe
  - `intrari` reprezintă descrierea, într-un format specific, a parametrilor de intrare ai rutinei. Pentru fiecare parametru trebuie precizate trei informaţii:
    * numele variabilei Scilab în care este memorată
    * poziţia (numărul de ordine) în lista de parametri ai rutinei
    * tipul: `'i'` – tip întreg sau caracter; `'r'` – tip real (simplă precizie); `'d'` – tip real (dublă precizie)
  - `'out'` reprezintă separatorul între lista de variabile de intrare şi lista de variabile de ieşire
  - `iesiri` reprezintă descrierea, într-un format specific, a parametrilor de ieşire ai rutinei. Pentru fiecare parametru de ieşire trebuie precizate trei informaţii:
    * dimensiunile matricei, sub forma:

      `[lin, col]`

      cu `lin` numărul de linii şi `col` numărul de coloane.
    * poziţia (numărul de ordine) în lista de parametri ai rutinei
    * tipul: `'i'` – tip întreg sau caracter; `'r'` – tip real (simplă precizie); `'d'` – tip real (dublă precizie)

Se constată că instrucţiunea de apel al unei rutine externe are o sintaxă relativ complicată, de aceea este de preferat ca pentru fiecare rutină externă să fie scrisă şi o funcţie Scilab care să o apeleze. Numele funcţiei Scilab poate fi acelaşi cu cel al rutinei externe.

Exemplu de apel al unor rutine externe

Să presupunem că rutinele externe, scrise în limbajul C, sunt următoarele:

- `suma` – calculează în variabila c suma a două numere reale, a şi b; funcţia este declarată sub forma:

```
void suma(float *a, float *b, float *c);
```

- `sumav` – calculează în variabila w suma, element cu element, a vectorilor u şi v; funcţia este declarată sub forma:

```
void sumav(int *n, float *u, float *v, float *w);
```

Să presupunem de asemenea că cele două rutine se găsesc în fişierul `test.o` plasat în directorul curent.

Următorul program Scilab realizează link–area rutinelor la Scilab şi apelul lor în Scilab.

```
// Apel functii suma, suma de vectori, scrise in C
ll = link('./test.o', 'suma', 'C');
ll = link('./test.o', 'sumav', 'C');

// Verificare: au fost linkate?
link()

// Citeste definitiile functiilor de apel numite la
//   fel ca si rutinele externe: suma, sumav
getf('functiiapel.sci');

// Apeleaza functiile
a = 1; b = 2;
c = suma(a,b)

u = 1:5; v = 1:5;
w = sumav(5,u,v)
```

Fişierul `functiiapel.sci` conţine definiţiile funcţiilor Scilab care apelează cele două rutine externe (observaţi folosirea caracterelor de continuare a comenzii ... pentru sporirea clarităţii):

```
function[c] = suma(a,b)
c = fort('suma',         ...    //nume rutina externa
          a,    1, 'r', ...    //intrari: nume a, par. 1, real
          b,    2, 'r', ...
    'out',               ...    //iesire: dimensiuni 1x1, par.3, real
          [1,1],3, 'r')
return

function[c] = sumav(n,a,b)
dim = size(a);
c = fort('sumav',     ...      //nume rutina externa
```

```
            n,  1, ´i´, ...      //intrari
            a,  2, ´r´, ...
            b,  3, ´r´, ...
      ´out´,            ...      //iesire
            dim,4, ´r´)
  return
```

6.11 Traducerea automată în FORTRAN a unei funcţii Scilab

După ce “prototipul” unui program a fost implementat şi testat în Scilab, de obicei, codul programului este tradus într-un alt limbaj de nivel înalt, cum ar fi C sau FORTRAN, în scopul sporirii eficienţei programului şi, dacă este cazul, al păstrării confidenţialităţii codului.

Scilab oferă facilitatea de a **traduce în mod automat** în FORTRAN o funcţie scrisă în Scilab.

Pentru aceasta trebuie parcurse următoarele etape (să presupunem că numele funcţiei este `func`):

- Se compilează funcţia:

  `comp(func)`

- Se defineşte o listă cu un număr de componente egal cu numărul parametrilor de **intrare** ai funcţiei:

  `l = list()`

  `l(1) = list('tip', 'nrlin', 'nrcol')`

  . . .

  unde fiecare componentă a listei caracterizează un parametru (în cazul general, acest parametru este o matrice) şi este la rândul său o listă cu trei elemente:

  – Tipul variabilei `tip`: `0` – întreg; `1` – real în dublă precizie; `10` – şir de caractere;
  – Numărul de linii `nrlin`: este un **nume formal** asociat numărului de linii ale variabilei, de exemplu `m` sau `nl`
  – Numărul de coloane `nrcol`: este un **nume formal** asociat numărului de coloane ale variabilei, de exemplu `n` sau `nc`

- Se apelează funcţia de traducere în FORTRAN:

  `mac2for( macr2lst(func), l)`

  Rezultatul apelului este un vector de şiruri de caractere, fiecare element al vectorului reprezentând o linie a programului FORTRAN.

Exemplu de utilizare a comenzii `mac2for`

Să presupunem că funcţia care trebuie tradusă în FORTRAN se găseşte în fişierul `sumaf.sci` şi are următoarea formă:

```
function[c] = sumaf(a,b)
// o functie simpla, pe care vrem sa o traducem in FORTRAN
// calculeaza suma matricelor (reale) a si b
  c = a+b
```

Următorul program Scilab realizează traducerea automată în FORTRAN a funcţiei `sumaf` şi scrie rezultatul (codul sursă al rutinei FORTRAN) în fişierul `sumaf.f`. Acest cod sursă poate fi ulterior compilat cu compilatorul FORTRAN şi linkeditat cu alte programe C, C++ sau FORTRAN.

```
// Citeste o functie Scilab si o traduce in FORTRAN
getf ('sumaf.sci')

// Functia sumaf are 2 par de intrare -- matrice de reali, a si b

l = list();
l(1) = list('1', 'm', 'n');
l(2) = list('1', 'm', 'n');

comp(sumaf)
txt = mac2for(macr2lst(sumaf), l);

// Scrie in fisier rezultatul
fis = file('open', 'sumaf.f', 'unknown')
write(fis, txt)
file('close', fis)
```

6.12 Exerciţii

EXERCIŢIUL 1

Scrieţi un program Scilab care să reprezinte grafic intensitatea curentului şi tensiunea la bornele condensatorului, în circuitul RC serie descris în Referatul Dvs. de laborator realizat la tema 4.

Programul va permite introducerea interactivă de la consolă a datelor de intrare – parametrii elementelor de circuit (R, C, E), va determina intervalul de timp şi pasul de

discretizare în timp adaptate valorilor parametrilor, şi va reprezenta grafic u_C şi i daţi de formulele analitice (paragraful 2 al referatului de laborator).

EXERCIŢIUL 2

Opţional: Salvaţi graficul în format XFig şi editaţi acest grafic, indicând prin săgeţi cele două grafice şi eventual adăugând alte elemente grafice pe care le consideraţi necesare pentru sporirea clarităţii graficului.

EXERCIŢIUL 3

a) Adăugaţi o nouă secţiune, **Rezultate numerice**, în referatul de laborator. În cadrul subsecţiunii **Implementarea în Scilab a formulelor analitice** descrieţi pe scurt funcţiile programului Scilab scris la exerciţiul 1 şi includeţi graficul realizat.

b) Adăugaţi încă o secţiune, **Codul sursă al programelor Scilab**, în partea de anexe. În subsecţiunea **Implementarea în Scilab a formulelor analitice** includeţi codul sursă al programului Dvs.

EXERCIŢIUL 4

a) Scrieţi un program Scilab care preia datele de intrare ale circuitului RC serie, stabileşte intervalul de timp adaptat datelor de intrare şi apoi foloseşte funcţiile Scilab de rezolvare a ecuaţiilor diferenţiale ordinare pentru a determina soluţia (tensiunea la bornele condensatorului). Programul va determina de asemenea intensitatea curentului în circuit, folosind tehnici de derivare numerică.

Refaceţi calculele pentru aceleaşi valori R, C şi E, dar pentru diverse valori ale parametrilor funcţiilor Scilab, respectiv pentru diverse valori ale pasului de discretizare temporală. Calculaţi, pentru fiecare set de valori ale parametrilor, eroarea relativă în norma euclidiană şi în norma Cebîşev între soluţiile numerice şi cele obţinute folosind formulele analitice. Salvaţi aceste valori într-un fişier ASCII, pentru a putea să le includeţi în referatul de laborator.

b) Pentru un set de valori ale parametrilor, reprezentaţi grafic în funcţie de timp tensiunea la bornele condensatorului (obţinută folosind formula analitică şi respectiv tehnica numerică). Reprezentaţi grafic intensitatea curentului (obţinută folosind formula analitică şi respectiv tehnica numerică).

c) Adăugaţi o nouă subsecţiune la secţiunea de **Rezultate numerice** a referatului de laborator. Includeţi: o scurtă descriere a funcţiilor programului Scilab şi a parametrilor acestuia, un tabel care să cuprindă erorile pentru diverse valori ale parametrilor, precum şi graficele curentului şi tensiunii.

d) Adăugaţi o nouă subsecţiune la anexa **Codul sursă al programelor Scilab**, numită **Programul de rezolvare cu Scilab a ecuaţiei diferenţiale ordinare**, în care includeţi codul sursă al programului scris.

Capitolul 7

Medii de programare Unix — C, C++, FORTRAN —

Oricât de performant şi de flexibil ar fi un pachet integrat specializat pentru aplicaţii ştiinţifice, matematice sau inginereşti, sunt situaţii în care cercetătorul este nevoit să facă apel la un limbaj universal de programare. Acest lucru se întâmplă atunci când:

- este necesară o mare **viteză** de execuţie (se poate spera cel puţin la înjumătăţirea timpului de calcul);
- este necesar un program **executabil** independent, care să fie distribuit în format binar (eventual păstrând confidenţial codul sursă);
- trebuie folosit un cod preexistent, eventual preluat dintr-o **bibliotecă matematică** standard.

În acest fel, pachetul integrat specializat devine un mediu pentru dezvoltarea (producerea şi testarea) rapidă a prototipurilor, fiind necesară ulterior traducerea manuală sau automată din limbajul specializat (de exemplu MATLAB) în limbajul universal.

Începând din anii '50, problema definirii unui limbaj universal pentru programarea calculatoarelor a generat pasionante studii şi discuţii în lumea specialiştilor în calculatoare. Cu toate că au fost definite sute sau poate chiar mii de limbaje, doar câteva au reuşti să se impună.

Dintre acestea, limbajul FORTRAN, lansat de firma IBM şi destinat iniţial calculelor matematice, reprezintă un exemplu uimitor de vitalitate. În ciuda vechimii sa, el este încă intens folosit de comunitatea ştiinţifică datorită cantităţii enorme de cod sursă FORTRAN de excepţională calitate (cum sunt bibliotecile matematice), şi datorită efortului de standardizare care urmăreşte permanent implementarea în limbaj a conceptelor moderne. Versiunile FORTRAN 77 [57] şi Fortran 90 [62] (apărute în 1977 şi respectiv 1990) au multe elemente diferite, progresul fiind evident.

Chiar dacă programele scrise în FORTRAN sunt relativ eficiente în execuţie, instrucţiunea de salt necondiţionat (`GOTO`) din acest limbaj a permis scrierea unor programe încâlcite ("spagheti"), greu de depanat şi întreţinut. Din acest motiv încă din anii '70 a apărut un curent care pledează pentru folosirea "**programării structurate**", bazată pe un număr redus de instrucţiuni de control (decizia şi ciclul).

S-a impus necesitatea utilizării unor tipuri de date clar definite (pentru a permite implementarea eficientă a "tipurilor abstracte de date") şi chiar apariţia unor limbaje de programare care să reflecte această viziune. Profesorul vienez Niklaus Wirth, autorul limbajului Pascal, a scris o carte fundamentală în domeniu, al cărei titlu "*Algorithms + Data Structures = Programs*" este ilustrativ în acest sens.

În paralel, în SUA, la laboratoarele Bell care aparţineau de AT&T, Dennis Ritchie a definit un nou limbaj universal numit de el **C** [61]. Acest limbaj, în ciuda aspectului său criptic (accesibil la prima vedere doar specialiştilor) s-a impus devenind limbajul oficial al sistemului de operare UNIX. De altfel, însuşi sistemul de operare UNIX este scris în limbajul **C** exceptând o parte nesemnificativă dependentă de maşină. Ca dovadă a succesului limbajului C s-a scris şi un translator automat din **FORTRAN** în **C**, numit `f2c`.

În afara modelului imperativ de programare structurată, cercetările de inteligenţă artificială au condus la definiţia unor limbaje funcţionale gen *Lisp,* potrivite pentru calcule simbolice, dar nu şi pentru cele numerice.

Folosirea intensivă a modelului imperativ a condus, la mijlocul anilor '80, la apariţia conceptului de programare orientată pe obiecte – **OOP**, concept fundamental pentru realizarea eficientă a programelor de mari dimensiuni, precum şi dezvoltarea de către Bjarne Stroustrup a unui limbaj numit **C++** [77]. Acesta este o extensie a limbajului **C** care permite implementarea caracteristicilor *OOP*: *încapsularea, moştenirea* şi *polimorfismul.*

Trebuie remarcat că realizarea unor programe de calitate este mai mult o problemă de stil de programare decât de limbaj. Se pot da exemple remarcabile [17] de coduri scrise în **C** sau chiar **FORTRAN** care satisfac cele mai exigente cerinţe, inclusiv cele ale orientării pe obiecte. Realizarea unor programe performante nu depinde nici măcar de calculatorul avut la dispoziţie, ci de calităţile programatorului. Nu trebuie uitat că sistemul de operare UNIX a fost dezvoltat pe un sistem DEC PDP care avea performanţe mai slabe decât orice calculator personal actual.

În consecinţă, vă recomandăm să vă perfecţionaţi stilul de programare, iar cea mai bună metodă este să studiaţi cu atenţie exemplele ilustre.

Compilatoarele sistemului Unix au proprietatea remarcabilă de **interoperabilitate**, ceea ce înseamnă că indiferent în ce limbaj a fost scris un program el poate apela rutine scrise în alt limbaj. În acest fel programatorii C pot exploatea tezaurul de cunoştinţe cuprins în bibliotecile matematice FORTRAN [74].

Cel mai mare depozit de software de calitate cu aplicaţii matematice şi ştiinţifice, disponibil în sursă (majoritatea FORTRAN) este NetLib [14], din care am selectat numele câtorva pachete celebre:

- `a` - algoritmi de aproximare
- `blas` - sistem algebric liniar de bază [45]
- `eispack` - vectori şi valori proprii pentru matrice [80]
- `fftpack` - transformanta Fourier rapidă
- `itpack` - rezolvarea iterativă a sistemelor liniare [48]
- `lapack` - algebră liniară
- `lanz` - valori proprii pentru matrice rare de mari dimensiuni
- `linpack` - sisteme de ecuaţii liniare
- `minpack` - sisteme de ecuaţii neliniare
- `odepack` - sisteme de ecuaţii diferenţiale ordinare
- `pppack` - subrutine de interpolare, spline
- `quadpack` - integrare numerică
- `sparspack` - algebră liniară cu matrice rare
- `specfun` - funcţii speciale
- `voronoi` - triangularizări Delaunay şi diagrame Voronoi

În acest de depozit se află sursele unor biblioteci complete cum sunt `sciport` (CRAY), `slatec` (Boeing) sau `toms` (ACM), precum şi pachete de funcţii ca `PVM`, `MPI`, `HENCE`, `PETSc` [17] ce permit dezvoltarea de algoritmi paraleli şi distribuiţi, care sunt executaţi simultan pe mai multe procesoare sau calculatoare cuplate în reţea.

Utilizarea în activitatea de cercetare a acestui tezaur de cunoştinţe vă va permite să eliminaţi greşeala de a reinventa roata şi să ridicaţi nivelul cercetărilor Dvs. la standardul mondial. Efortul de a învăţa să folosiţi ce au realizat alţii va fi răsplătit însutit!

Obiectivul capitolului de faţă nu îl constituie învăţarea unui limbaj de programare (de altfel nici nu ar fi posibil în câteva ore!) ci să învăţaţi folosirea mediului de dezvoltare sub UNIX specific LMN (editare, compilare, linkeditare, depanare); presupunând că stăpâniţi un limbaj de programare de nivel înalt (FORTRAN, C sau C++). În plus, se urmăreşte însuşirea deprinderilor minimale necesare folosirii unei subrutine din biblioteca matematică FORTRAN, cu apel din C. În acest fel vi se deschid porţile unui univers extrem de bogat, cuprins în biblioteci de renume.

Pentru informaţii suplimentare referitoare la limbajul C, se pot consulta referinţele [46], [70] şi [79] pp. 315–333, pentru traducerea din pseudocod în C, C++ şi FORTRAN vă recomandăm [59] pp. 261–322, iar pentru limbajul FORTRAN vă recomandăm [57] şi [62].

7.1 Principalele opţiuni ale compilatoarelor `gcc`, `g++`, `g77`

Compilatoarele `gcc`, `g++`, `g77` sunt versiuni GNU ale compilatoarelor C, C++ şi respectiv FORTRAN 77 în mediul Unix.

De fapt, compilatoarele sunt integrate, orice fişier sursă putând fi compilat cu compilatorul `gcc` care identifică tipul fişierului corespunzător extensiei (`.c`, `.cc` şi respectiv `.f`) şi interpretează corespunzător instrucţiunile din fişierul sursă.

În consecinţă, opţiunile de compilare de bază sunt identice pentru toate cele trei compilatoare.

`-o` *fis_ex* – fişierul executabil creat de compilator să aibă numele *fis_ex* (implicit: `a.out`)

`-g` – fişierul executabil să conţină informaţii necesare pentru depanarea programului (nume variabile şi plasarea lor în memorie, adresele liniilor de cod sursă)

`-c` – nu parcurge decât etapa de compilare nu şi cea de linkeditare; rezultatul este un fişier obiect (cu extensia `.o`)

`-O`*n* – compilează cu optimizare; numărul *n* este opţional şi poate lua valorile 0 (fără optimizare), 1 (implicit), 2 sau 3

`-p` fişierul executabil să includă instrucţiuni suplimentare pentru generarea, după rularea sa, a unei liste de informaţii despre program (*program profile*), cum ar fi: număr de apeluri ale funcţiilor, timpul consumat cu execuţia fiecărei funcţii, etc.

`-l`*nume* – linkeditare cu biblioteca matematică `lib`*nume*.

Observaţie: În mediul Unix, apelul compilatorului determină atât compilarea fişierelor sursă cât şi linkeditarea fişierelor cu alte fişiere obiect sau cu biblioteci standard.

De exemplu:

```
gcc -g -o test -p  test.c -lm
```

are ca efect compilarea sursei `test.c` cu includerea: informaţiilor pentru depanare (`-g`), a informaţiilor de "profile" (`-p`), scrierea fişierului executabil sub numele `test` şi linkeditarea cu biblioteca matematică `libm.a`.

Pentru a nu trebui să introduceţi de fiecare dată o comandă de compilare complicată, puteţi crea un script care să conţină instrucţiunea de compilare. În cazul programelor ale căror surse se află în mai multe fişiere, este recomandabil să folosiţi comanda `make` (paragraful 7.3).

Extensii fişiere:

- `.c` – fişiere sursă în limbaj C
- `.cc` – fişiere sursă în limbaj C++
- `.f` – fişiere sursă în limbaj FORTRAN77
- `.h` – fişiere de tip "include"
- `.o` – fişiere obiect (rezultate în urma compilării, înaintea creării fişierului executabil)

7.2 Recomandări privind stilul de lucru profesional

Productivitatea şi eficienţa programării în mediul de programare Unix depind în mod esenţial de folosirea "inteligentă" a facilităţilor existente.

Deşi fiecare programator are propriile tehnici de creştere a eficienţei, atunci când sunteţi la început puteţi folosi sugestiile de mai jos.

- Cum lucraţi într-un mediu care permite deschiderea mai multor ferestre, folosiţi această facilitate!

  Utilizaţi o fereastră pentru editarea programului, altă fereastră pentru compilări şi, de preferinţă, o a treia fereastră pentru execuţia programului.

  După ce aţi terminat de editat programul, salvaţi-l, dar nu ieşiţi din editor: în cazul în care programul Dvs. ar avea o eroare pe care ar trebui să o corectaţi, veţi economisi astfel timpul de lansare, încă o dată, a editorului.

- Utilizaţi de preferinţă editorul Emacs pentru editarea programelor. Acest editor are facilitatea de a lansa comanda de compilare, fără măcar să trebuiască să mutaţi mouse–ul în altă fereastră!

  La programe lungi, evitarea chiar şi a unei simple operaţii precum mutarea în altă fereastră va contribui la micşorarea timpului de editare şi depanare a programului.

- Dacă aveţi de realizat un program asemănător unuia existent, nu îl scrieţi de la zero! Copiaţi programul existent sub numele dorit pentru programul Dvs. şi modificaţi-l.

  Această tehnică, de folosire a unor *modele* (patterns) preexistente (preluate din fişierele de exemple, tutorials) vă poate reduce substanţial timpul de introducere a programului şi numărul de erori de programare.

- Rareori vi se va întâmpla să scrieţi un program, să îl depanaţi şi să îl executaţi, iar apoi să nu îl mai "atingeţi" niciodată. De cele mai multe ori, un program odată scris trece prin multe modificări, unele dintre ele făcute la mult timp după data primei creări. Uneori, nu programatorul iniţial este cel care modifică programul, ci un alt membru al echipei.

Pentru a putea reduce sensibil efortul de întreţinere, documentaţi programul încă de la început. Altfel veţi irosi un timp enorm de descifrare atunci când ulterior trebuie să faceţi o mică modificare.

Comentaţi programele scrise: începeţi cu un comentariu privind scopul programului, numele programatorului, data la care a fost realizat.

Începeţi fiecare rutină cu un comentariu privind scopul funcţiei şi semnificaţiile parametrilor.

Adăugaţi comentarii şi la acele linii de program care codifică o metodă neobişnuită de realizare a unei operaţii, la acelea foarte "concentrate", care conţin diferite artificii de programare, etc.

După ce aţi terminat de scris şi de depanat programul, scrieţi şi o documentaţie a acestuia, de tip "Manualul utilizatorului" şi chiar "Manualul programatorului" în care descrieţi tehnicile de programare folosite (algoritmi şi structuri de date). În 90% din cazuri, după un timp vă veţi felicita că aţi făcut acest efort.

7.3 Compilarea programelor folosind `make`

Un program de mari dimensiuni, al cărui cod sursă se află în mai multe fişiere separate, poate necesita un timp foarte lung pentru compilare şi editarea legăturilor necesare constituirii programului executabil.

Programul utilitar `make` determină automat care părţi ale unui astfel de program s-au modificat şi trebuie recompilate, "comandă" recompilarea selectivă şi apoi linkeditarea.

Pentru a putea utiliza comanda `make` utilizatorul trebuie să creeze un fişier numit `Makefile` sau `makefile`, în care sunt "descrise", într-un format specific, relaţiile dintre fişierele care compun programul precum şi comenzile care realizează compilarea.

7.3.1 Structura unui fişier `makefile`

Un fişier `makefile` conţine următoarele tipuri de linii:

- comentarii: linii care încep cu caracterul `#`
- definiţii de variabile: `VARIABILA = valoare`

  Ex: `CFLAGS = -g -c`

  O variabilă definită poate fi folosită în liniile următoare ale fişierului `makefile`, sub forma

  ```
  $(VARIABILA)
  ```

- definiţii de relaţii:

  `fişier:  listă_fişiere`

  în care, în cazul nostru, `fişier` este numele unui fişier obiect (cu extensia `.o`) iar `listă_fişiere` este o listă de fişiere de tip `.c` şi `.h` de care "depinde" fişierul obiect. De exemplu, dacă un modul al programului se numeşte `test.c` şi el include fişierul header `test.h` (conţine instrucţiunea `#include "test.h"`) atunci: fişierul obiect `test.o` trebuie re–generat dacă se modifică `test.c` sau se modifică `test.h`. Această dependenţă va apărea în fişierul `makefile` sub forma:

  `test.o:  test.c test.h`

- instrucţiuni de generare a principalului "rezultat"

  Sunt linii care încep cu Tab, plasate imediat sub o linie de tip "definire de relaţie", care conţin instrucţiunea care "generează" obiectul.

  Exemplu:

```
test.o: test.c test.h}
        gcc $(CFLAGS) test.c
```

7.3.2 Modele de fişiere makefile

Iată mai jos un **model simplu** de fişier `makefile`. Acest model presupune următoarele:

- Codul sursă al programului se află în două fişiere: `inout.c` şi `main.c`.

  Fişierul `inout.c` include fişierul header `inout.h`, iar fişierul `main.c` include fişierele header `inout.h` şi `main.h`.

- Fişierul executabil trebuie salvat sub numele `calcul`.

```
main: main.o inout.o
      gcc -g -p main.o inout.o -o calcul -lm

# Dependencies
main.o: main.c inout.h main.h
      gcc -g -p -c main.c -o main.o
inout.o: inout.c inout.h
      gcc -g -p -c inout.c -o inout.o
```

Modelul de mai sus, extrem de simplu, poate fi dezvoltat astfel încât programul scris să fie uşor de întreţinut, prin salvarea tuturor fişierelor în directoare dedicate.

Vă propunem un **model** de fişier `makefile` pe care vă recomandăm să îl folosiţi, adaptându-l programului concret pe care îl scrieţi.

Acest model presupune următoarele:

- În directorul curent există patru subdirectoare: `include` (conţine fişierele de tip `.h`); `src` (conţine fişierele de tip `.c`); `obj` (conţine fişierele de tip `.o`); `bin` (conţine fişierele executabile).
- Codul sursă al programului se află în două fişiere: `inout.c` şi `main.c`, plasate în subdirectorul `src`.

  Fişierul `inout.c` include fişierul header `inout.h`, iar fişierul `main.c` include fişierele header `inout.h` şi `main.h`. Fişierele header sunt plasate în subdirectorul `include`.
- Pentru a păstra "ordinea" în directoarele `include` şi `src`, se doreşte salvarea fişierelor obiect în directorul `obj`.
- Fişierul executabil trebuie salvat în directorul `bin` sub numele `calcul`.

O posibilă structură a fişierului `makefile` este următoarea:

```
CC = gcc
CFLAGS = -g -p
INCLUDEFLAGS = -Iinclude

INCLUDE_DIR = include
SRC_DIR = src
OBJ_DIR = obj
BIN_DIR = bin
PROGNAME = calcul
LIBS = -lm

OBJECTS = $(OBJ_DIR)/main.o \
          $(OBJ_DIR)/inout.o

main: $(OBJECTS)
      $(CC) $(CFLAGS) $(OBJECTS) $(LIBS) -o $(BIN_DIR)/$(PROGNAME)

# Dependencies
$(OBJ_DIR)/main.o: $(SRC_DIR)/main.c $(INCLUDE_DIR)/inout.h $(INCLUDE_DIR)/main.h
      $(CC) $(CFLAGS) $(INCLUDEFLAGS) -c $(SRC_DIR)/main.c -o $(OBJ_DIR)/main.o
$(OBJ_DIR)/inout.o: $(SRC_DIR)/inout.c $(INCLUDE_DIR)/inout.h
      $(CC) $(CFLAGS) $(INCLUDEFLAGS) -c $(SRC_DIR)/inout.c -o $(OBJ_DIR)/inout.o
```

7.4 Utilizarea depanatorului gdb

Lansare:

`gdb <↵ >`

sau, pentru versiunea X:

`xxgdb <↵ >`

- `file` *nume*: încărcarea fişierului **executabil** *nume*
- `set args` *argumente*: definirea argumentelor programului (cele care ar apărea în linia de comandă, dacă programul ar fi lansat de la consolă şi nu din debugger)
- `break` *numar*: setare breakpoint (oprire) pe linia cu numărul *numar* din fişierul **sursă** (*.c)
- `display` *var*: afişarea valorii variabilei *var*
- `watch` *var*: afişarea valorii variabilei *var* automat, de câte ori aceasta îşi schimbă valoarea.
- `run`: rularea programului
- `continue`: continuarea execuţiei programului după o oprire
- `step`: rularea programului pas cu pas
- `next`: rularea programului pas cu pas, fără a intra în alte funcţii
- `kill`: oprirea execuţiei programului
- `quit`: ieşirea din `gdb`
- `list`: afişarea a 10 linii din program

Observaţie: `xxgdb` este o versiune a `gdb` bazată pe sistemul de ferestre grafice XWindow. Toate comenzile `gdb` sunt valabile şi în `xxgdb`. În plus, `xxgdb` permite selectarea majorităţii comenzilor `gdb` şi cu ajutorul mouse-ului.

Fereastra programului `xxgdb` este prezentată în figura 7.1.

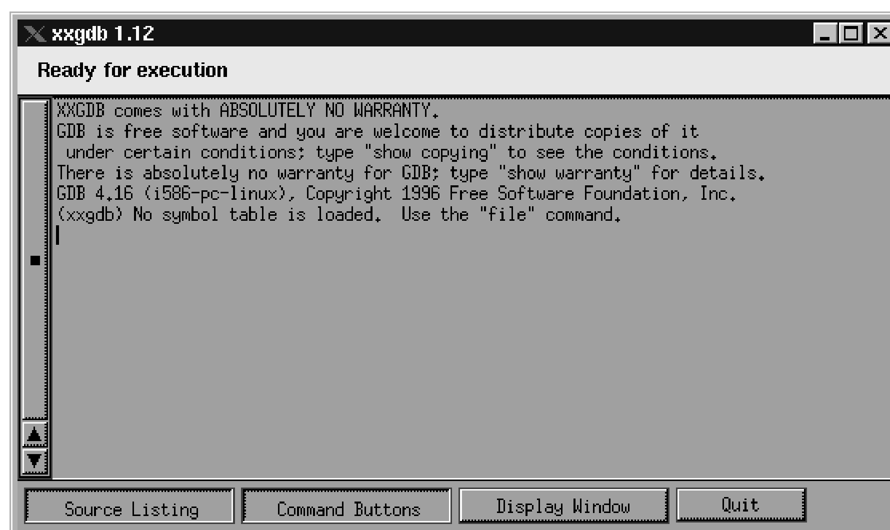


Fig. 7.1: Fereastra principală a programului `xxgdb`

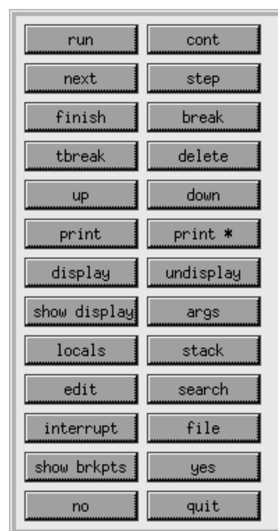


Fig. 7.2: Butoanele de comenzi ale programului `xxgdb`

După apăsarea butonului "Command Buttons" al ferestrei principale, se afişează fereastra de butoane, al cărei aspect este prezentat în figura 7.2.

Pentru informaţii suplimentare referitoare la debugger-ele `gdb` şi `xxgdb`, se pot consulta referinţele [58] pp. 489–497 şi [79] pp. 355–371.

7.5 Apelarea rutinelor FORTRAN dintr-un program scris în limbajul C

Pentru a apela o rutină FORTRAN dintr-un program C, trebuie să se ţină cont de următoarele observaţii:

- Toţi parametrii unei rutine FORTRAN sunt transmişi prin adresă. În consecinţă: tablourile (vectori şi matrice), şirurile de caractere şi funcţiile se transmit ca parametri spre o rutină FORTRAN la fel ca şi în C. În schimb, scalarii trebuie transmişi prin adresă.

  De exemplu: dacă `i` este o variabilă declarată prin instrucţiunea

  ```
  int i;
  ```

  ea va fi transmisă ca parametru unei funcţii FORTRAN sub forma

  ```
  &i
  ```

- Stocarea în memorie a matricelor este făcută în mod diferit de modul de stocare C, şi anume: întâi pe linii şi apoi pe coloane (în ordinea: $a_{11}, a_{21}, \ldots a_{n1}, a_{12}, a_{22}, \ldots$.

  De aceea, o matrice care urmează să fie transmisă ca parametru unei rutine FORTRAN trebuie permutată înainte de a apela funcţia FORTRAN.

- Transmiterea unei variabile de tip complex se face astfel:

  - Variabila se declară sub forma:

    ```
    struct complex {float re; float im;} varc;
    ```

  - Transmiterea se face prin adresa variabilei `varc`:

    ```
    &varc
    ```

Exemplu

Se consideră următorul antet al unei funcţii FORTRAN:

```
c FUNCTIA    calcul
c Parametri de intrare:
c     f: subrutina, definita ca:
c             subroutine f(a,b,c,n)
c             real b(10),c(10), a(10,10)
c             integer n
c     g, h: variabile de tip real
c     titlu: sir de 80 caractere
c     a: matrice cu 8 linii si 10 coloane
c     b, c: vectori cu 10 componente
c     n: intreg
```

```
c Parametri de iesire:
c     gh: variabila de tip real
c     ierr: intreg
c     cx: complex
c Functia intoarce:
c     o variabila de tip intreg

      integer function calcul(f,g,h,gh,titlu,a,b,c,n,ierr,cx)

      external f
      real g, h, gh
      character*80 titlu
      real b(10), c(10), a(8,10)
      complex cx
```

Această funcţie poate fi apelată din FORTRAN, folosind următorul program principal:

```
c Program FORTRAN de apel al rutinei "calcul"
      program main

      external f
      real g, h, gh
      character*80 titlu
      real b(10), c(10), a(10,8)
      integer ii
      complex cx;

      titlu = ´da´
      n = 2
      a(1,1) = 1
      a(2,1) = 2
      a(1,2) = 3
      a(2,2) = 4

      b(1) = 1
      b(2) = 2

      g = 7
      h = 8
      ii = calcul(f,g,h,gh,titlu,a,b,c,n,ierr,cx)

      write(*,*) ´c = ´, c
      write(*,*) ´gh = ´, gh
      write(*,*) ´titlu = ´,  titlu
      write(*,*) ´ierr = ´, ierr
      write(*,*) ´ii = ´, ii
      write(*,*) ´cx = ´, cx
      end
```

Programul C care apelează această funcţie are forma:

```
/* Program C de apel al functiei "calcul" */
#include <stdlib.h>
#include <string.h>

struct complex {float re; float im;};

void f(float a[8][10], float b[10],float c[10],int n);
int calcul(void f(float a[8][10], float b[10],float c[10],int n),
        float * g, float * h, float * gh,
        char *titlu, float a[10][8], float b[10],float c[10],int * n,
        int * ierr, struct complex  * cx );

main() {
      float g, h, gh;
      char titlu[80];
      float b[10], c[10], a[8][10];
      int ii, ierr, n, len;
      struct complex cx;

      strcpy(titlu, "da");

      /* Matrice cu elementele:
         | 1 3 |
         | 2 4 |
      transpusa pentru a fi transmisa ca parametru */
      n = 2;
      a[0][0] = 1; a[0][1] = 2;
      a[1][0] = 3; a[1][1] = 4;

      b[0] = 1; b[1] = 2;
      g = 7.0; h = 8.0;

      ii = calcul(f,&g,&h,&gh,titlu,a,b,c,&n,&ierr,&cx);

      printf("c = %f %f %f \n", c[0], c[1], c[2]);
      printf("gh = %f \n", gh);
      printf("titlu = %s \n", titlu);
      printf("ierr = %d \n", ierr);
      printf("ii = %d \n", ii);
      printf("cx = %f %f \n", cx.re, cx.im);
} /* End main */
```

7.6 Exerciţii

EXERCIŢIUL 1

Primul program

1. Creaţi un director numit ~/programe
2. Folosind editorul vi sau **emacs** introduceţi următorul program (în directorul ~/programe):

```
main()
{
float a, b, c;

a = 2.;
b = 3.;

c = a + b;
printf("Suma este: %f.\n", c);

c = a * b;
printf("Produsul este: %d.\n", c);

}        /* end main */
```

3. Salvaţi fişierul sub numele test.c.

   Vă recomandăm să NU ieşiti din editor, ci să deschideţi o nouă fereastră în care veţi da comenzile de compilare şi veţi executa programul.
4. Compilaţi programul folosind comanda:

   `gcc -g test.c`
5. Apelând ls, veţi constata că s-a creat fişierul executabil a.out.
6. Executaţi programul, cu comanda:

   `a.out`
7. Observaţi efectul formatului %d (cu semnificaţia: valoare de tip întreg) asupra rezultatului afişat.

Note:

- Orice program C trebuie să conţină cel puţin o funcţie, funcţia main.

- comentariile sunt delimitate de caracterele `/* */`
- Principalele tipuri de date sunt: caracter (`char`), întreg (`int`), real (`float`).
- Orice instrucţiune se încheie cu ; (punct şi virgulă)
- Funcţia `printf` afişează pe ecran un text.

  Instrucţiunea: `printf("Suma este: %f.\n", c);` are următorul efect: se afişează pe ecran textul `Suma este:` , urmat de valoarea unei variabile de tip "float" (formatul "float" este desemnat prin `%f`), urmat de punct şi de trecerea la linie nouă `\n`. Şirul de caractere delimitat de `"` este primul parametru al funcţiei `printf`. Următorii parametri (în cazul de faţă unul singur) reprezintă variabilele ale căror valori vor fi afişate, conform formatului specificat în primul parametru.

EXERCIŢIUL 2

Primul program, extins

1. Lucrând în fereastra de editare a programului, modificaţi programul `test.c` astfel încât să preia de la consolă valorile parametrilor `a, b`. Pentru aceasta:
   - Ştergeţi cele două linii care setează valorile parametrilor `a` şi `b`.
   - Introduceţi în locul lor liniile:

   ```
   printf("Introduceti primul numar: ");
   scanf("%d", &a );

   printf("Introduceti al doilea numar: ");
   scanf("%f", &a );
   ```

2. Folosind fereastra pe care aţi dedicat-o operaţiunilor de compilare şi execuţie a programului, compilaţi din nou si executaţi programul, dând valorile $a = 1.5$ şi $b = 2.0$. Veti constata că rezultatul nu este cel aşteptat (deci, există o greşeală de logică în program). Va trebui, dacă nu vă daţi seama direct, să utilizaţi un "depanator" (debugger) pentru a depista eroarea.

EXERCIŢIUL 3

Depanarea folosind debugger-ul `xxgdb`

1. Lansaţi debugger-ul folosind butonul din stânga al mouse-ului şi selectând Applications → xxgdb. Alternativ, îl puteţi lansa din linia de comandă, tastând:

   ```
   xxgdb <Enter>
   ```

2. Deschideţi fereastra pentru codul sursă, fereastra de butoane şi fereastra "Display".

3. Selectaţi "File" din fereastra de butoane şi apoi alegeţi din lista de fişiere afişată fişierul executabil `a.out`.

4. Setaţi un "breakpoint" (punct de oprire) pe linia

   ```
   printf("Introduceti al doilea numar:  ");
   ```

   astfel:

   - În fereastra conţinând textul sursă, deplasaţi cursorul pe linia respectivă şi selectaţi "Break" din fereastra de butoane

     sau

   - În fereastra iniţială, daţi comanda:

     `b` <*număr*>

     unde *număr* este numărul liniei pe care se doreşte oprirea, afişat în fereastra conţinând textul sursă în partea din dreapta sus.

5. Selectaţi "run" (sau daţi comanda `r`). Programul se va executa până la linia marcată ca "breakpoint".

6. Daţi comanda de afişare a valorii variabilei `a`:

   ```
   p a
   ```

   Alternativ, cu comanda `display a`, variabila se va afişa în fereastra "Display".

7. Încercaţi să vă daţi seama unde poate fi greşeala din linia anterioară celei pe care rularea s-a oprit.

8. Setaţi un alt breakpoint pe linia:

   ```
   c = a + b;
   ```

9. Continuaţi rularea (cu una din comenzile: Step, Run sau Continue) şi după oprire verificaţi valoarea variabilei `b`. Încercaţi să depistaţi cea de a doua eroare.

10. În fine, setaţi un alt breakpoint pe linia:

    ```
    printf("Produsul este:  %d.\n", c);
    ```

    şi încercaţi să depistaţi din ce motiv rezultatul afişat (care ar trebui să fie produsul `a * b`) nu este corect.

EXERCIŢIUL 4

Citiţi cu atenţie următorul cod cu scopul de a înţelege fiecare instrucţiune. Ce se va afişa la consolă?

```
float v[50], u[50];        /* Doi vectori */
int n=10;                  /* Un intreg initializat cu 10 */

for (i=0, j=0;   i<n;  i++, j=j+2)
    {
    v[i] = i;
    u[i] = j;
    printf("v[%d] = %f, u[%d] = %f \n", i, v[i], i, u[i]);
    }
```

EXERCIŢIUL 5

Scrieţi un program care să calculeze produsul scalar a doi vectori.

- Dimensiunea maximă a vectorilor: 20
- Dimensiunea actuală, precum şi componentele vectorilor vor fi introduse de la consolă.
- Programul va conţine două funcţii:
  1. `main`: preia valorile de intrare, apelează `prod_s`, afişează rezultatul;
  2. `prod_s`: primeşte ca parametri doi vectori şi returnează rezultatul (produsul scalar) în cel de al treilea parametru.
- Cele două funcţii vor fi scrise în fişiere separate, `main.c` şi respectiv `prods.c`.
- Folosiţi utilitarul `make` pentru compilare şi editarea legăturilor.

EXERCIŢIUL 6

Folosind facilitatea Scilab de a apela rutine externe (paragraful 6.10), incorporaţi în Scilab, executaţi şi verificaţi în mediul Scilab rutina de calcul al produsului scalar.

EXERCIŢIUL 7

Folosind facilitatea Scilab `mac2for` (paragraful 6.11) generaţi automat codul FORTRAN al următoarei funcţii Scilab:

```
function[b,err] = permuta(a)
// permuta elementele vectorului a, astfel incat:
//   ultimul devine primul, penultimul devine al doilea, etc.
  [l,c] = size(a);
  if l ~= 1 then
```

```
    err = 1;
    b = a;
    return;
  end
  b = a(c:-1:1);
  err = 0
```

Compilaţi şi executaţi acest cod, atât în Scilab cât şi în FORTRAN. Pentru a testa codul FORTRAN, scrieţi un program principal în C care să preia parametrii de intrare şi să apeleze rutina `permuta`.

EXERCIŢIUL 8

Folosind comanda `f2c`, traduceţi automat în C programul FORTRAN generat la exerciţiul 7. Compilaţi programul rezultat şi verificaţi rezultatul.

EXERCIŢIUL 9

Scrieţi un raport (referat de laborator) care să descrie modul de rezolvare a exerciţiilor anterioare şi rezultatele obţinute. Comentaţi dificultăţile apărute. Anexaţi codul sursă al tuturor programelor scrise, precum şi conţinutul fişierelor `Makefile` folosite.

Următoarele exerciţii sunt facultative. Vă recomandăm să le faceţi, deoarece scopul lor este de a vă familiariza cu o facilitate extrem de utilă pentru productivitatea muncii Dvs. de cercetare: utilizarea rutinelor din bibliotecile matematice profesionale.

EXERCIŢIUL 10

Scrieţi un program principal în limbajul C care apelează o rutină din biblioteca matematică FORTRAN.

Rutina apelată este `QAGI` şi face parte din pachetul de rutine `QUADPACK` dedicat integrării numerice [19]. Codul sursă al acestui pachet este depozitat în biblioteca matematică Netlib [14].

Antetul rutinei QAGI este:

```
      subroutine qagi(f,bound,inf,epsabs,epsrel,result,abserr,neval,
     *   ier,limit,lenw,last,iwork,work)
c***begin prologue  qagi
c***date written   800101   (yymmdd)
c***revision date  830518   (yymmdd)
c***category no.  h2a3a1,h2a4a1
c***keywords  automatic integrator, infinite intervals,
```

```
c             general-purpose, transformation, extrapolation,
c             globally adaptive
c***author  piessens,robert,appl. math. & progr. div. - k.u.leuven
c           de doncker,elise,appl. math. & progr. div. -k.u.leuven
c***purpose  the routine calculates an approximation result to a given
c            integral   i = integral of f over (bound,+infinity)
c                    or i = integral of f over (-infinity,bound)
c                    or i = integral of f over (-infinity,+infinity)
c            hopefully satisfying following claim for accuracy
c            abs(i-result).le.max(epsabs,epsrel*abs(i)).
c***description
c
c        integration over infinite intervals
c        standard fortran subroutine
c
c        parameters
c         on entry
c            f      - real
c                     function subprogram defining the integrand
c                     function f(x). the actual name for f needs to be
c                     declared e x t e r n a l in the driver program.
c
c            bound  - real
c                     finite bound of integration range
c                     (has no meaning if interval is doubly-infinite)
c
c            inf    - integer
c                     indicating the kind of integration range involved
c                     inf = 1 corresponds to  (bound,+infinity),
c                     inf = -1             to  (-infinity,bound),
c                     inf = 2              to (-infinity,+infinity).
c
c            epsabs - real
c                     absolute accuracy requested
c            epsrel - real
c                     relative accuracy requested
c                     if  epsabs.le.0
c                     and epsrel.lt.max(50*rel.mach.acc.,0.5d-28),
c                     the routine will end with ier = 6.
c
c
c         on return
c            result - real
c                     approximation to the integral
c
c            abserr - real
c                     estimate of the modulus of the absolute error,
c                     which should equal or exceed abs(i-result)
```

```
c
c          neval  - integer
c                   number of integrand evaluations
c
c          ier    - integer
c                   ier = 0 normal and reliable termination of the
c                           routine. it is assumed that the requested
c                           accuracy has been achieved.
c                 - ier.gt.0 abnormal termination of the routine. the
c                           estimates for result and error are less
c                           reliable. it is assumed that the requested
c                           accuracy has not been achieved.
c          error messages
c                   ier = 1 maximum number of subdivisions allowed
c                           has been achieved. one can allow more
c                           subdivisions by increasing the value of
c                           limit (and taking the according dimension
c                           adjustments into account). however, if
c                           this yields no improvement it is advised
c                           to analyze the integrand in order to
c                           determine the integration difficulties. if
c                           the position of a local difficulty can be
c                           determined (e.g. singularity,
c                           discontinuity within the interval) one
c                           will probably gain from splitting up the
c                           interval at this point and calling the
c                           integrator on the subranges. if possible,
c                           an appropriate special-purpose integrator
c                           should be used, which is designed for
c                           handling the type of difficulty involved.
c                       = 2 the occurrence of roundoff error is
c                           detected, which prevents the requested
c                           tolerance from being achieved.
c                           the error may be under-estimated.
c                       = 3 extremely bad integrand behaviour occurs
c                           at some points of the integration
c                           interval.
c                       = 4 the algorithm does not converge.
c                           roundoff error is detected in the
c                           extrapolation table.
c                           it is assumed that the requested tolerance
c                           cannot be achieved, and that the returned
c                           result is the best which can be obtained.
c                       = 5 the integral is probably divergent, or
c                           slowly convergent. it must be noted that
c                           divergence can occur with any other value
c                           of ier.
c                       = 6 the input is invalid, because
```

```
c                              (epsabs.le.0 and
c                               epsrel.lt.max(50*rel.mach.acc.,0.5d-28))
c                               or limit.lt.1 or leniw.lt.limit*4.
c                              result, abserr, neval, last are set to
c                              zero. exept when limit or leniw is
c                              invalid, iwork(1), work(limit*2+1) and
c                              work(limit*3+1) are set to zero, work(1)
c                              is set to a and work(limit+1) to b.
c
c          dimensioning parameters
c             limit - integer
c                     dimensioning parameter for iwork
c                     limit determines the maximum number of subintervals
c                     in the partition of the given integration interval
c                     (a,b), limit.ge.1.
c                     if limit.lt.1, the routine will end with ier = 6.
c
c             lenw  - integer
c                     dimensioning parameter for work
c                     lenw must be at least limit*4.
c                     if lenw.lt.limit*4, the routine will end
c                     with ier = 6.
c
c             last  - integer
c                     on return, last equals the number of subintervals
c                     produced in the subdivision process, which
c                     determines the number of significant elements
c                     actually in the work arrays.
c
c          work arrays
c             iwork - integer
c                     vector of dimension at least limit, the first
c                     k elements of which contain pointers
c                     to the error estimates over the subintervals,
c                     such that work(limit*3+iwork(1)),... ,
c                     work(limit*3+iwork(k)) form a decreasing
c                     sequence, with k = last if last.le.(limit/2+2), and
c                     k = limit+1-last otherwise
c
c             work  - real
c                     vector of dimension at least lenw
c                     on return
c                     work(1), ..., work(last) contain the left
c                      end points of the subintervals in the
c                      partition of (a,b),
c                     work(limit+1), ..., work(limit+last) contain
c                      the right end points,
c                     work(limit*2+1), ...,work(limit*2+last) contain the
```

```
c                      integral approximations over the subintervals,
c                     work(limit*3+1), ..., work(limit*3)
c                      contain the error estimates.
c***references  (none)
c***routines called  qagie,xerror
c***end prologue  qagi
```

Programul scris de Dvs. trebuie să fie traducerea în limbajul C a programului de test FORTRAN:

```
C 4.5. Calling Program for QAGI
C      ------------------------
C
C            REAL ABSERR,BOUN,EPSABS,EPSREL,F,RESULT,WORK
C            INTEGER IER,INF,IWORK,LAST,LENW,LIMIT,NEVAL
C            DIMENSION IWORK(100),WORK(400)
C            EXTERNAL F
C            BOUN = 0.0E0
C            INF = 1
C            EPSABS = 0.0E0
C            EPSREL = 1.0E-3
C            LIMIT = 100
C            LENW = LIMIT*4
C            CALL QAGI(F,BOUN,INF,EPSABS,EPSREL,RESULT,ABSERR,NEVAL,
C           *  IER,LIMIT,LENW,LAST,IWORK,WORK)
C      C  INCLUDE WRITE STATEMENTS
C            STOP
C            END
C      C
C            REAL FUNCTION F(X)
C            REAL X
C            F = 0.0E0
C            IF(X.GT.0.0E0) F = SQRT(X)*ALOG(X)/
C           *                   ((X+1.0E0)*(X+2.0E0))
C            RETURN
C            END
C
```

Programul de mai sus realizează calculul numeric al integralei:

$$\int_0^\infty \frac{\sqrt{x}\ln x}{(x+1)(x+2)} = \pi\sqrt{2}\ \ln 2 \approx 3.0795717821424 \tag{7.1}$$

De această dată veţi primi prin poştă electronică de la instructor codul sursă al subrutinei, dar în viitor va trebui să vă familiarizaţi cu explorarea Netlib şi cu teleîncărcarea prin Internet a subrutinei potrivite.

Indicaţii

- Declaraţiile datelor vor fi traduse în C astfel:

  `INTEGER` → `int`

  `REAL` → `float`

  `DIMENSION` → tipul variabilei este stabilit printr-o declaraţie de tip anterioară, instrucţiunea `DIMENSION` precizând doar faptul că datele sunt vectori sau matrice şi dimensiunile acestora.

- Eventualele variabile nedefinite în sursele FORTRAN se consideră declarate implicit prin prima literă a numelui lor, indiferent că sunt nume de variabile, tablouri sau funcţii.

  Convenţia de declarare implicită este următoarea: variabilele al căror nume începe cu una din literele `i, j, k, l, m, n` sunt de tip întreg, toate celelalte sunt implicit de tip real.

- Cuvântul cheie `EXTERNAL` reprezintă o declaraţie de funcţie. În FORTRAN nu este necesar să se precizeze în declaraţia unei funcţii `EXTERNAL` şi parametrii ei.

  În C, nu există un cuvânt cheie corespunzător lui `EXTERNAL`, în schimb, va trebui să declaraţi funcţia (tipul, numele, lista de parametri cu tipurile lor).

- Cuvântul cheie `CALL` este folosit în FORTRAN pentru apelul subrutinelor.

  În C, nu există un cuvânt cheie special, ci apelul unei rutine care nu întoarce o valoare (funcţie C de tip `void`) este făcut doar prin invocarea numelui urmat de o listă a parametrilor actuali.

- Atenţie la definirea parametrilor funcţiei! Înainte de a o scrie, revedeţi paragraful 7.5.

- Cuvintele cheie `STOP` şi `END` din FORTRAN semnifică sfârşitul programului principal, şi sunt echivalente acoladei închise (}) de sfârşit al funcţiei `main` din programul C.

- Spre deosebire de C, variabilele şi instrucţiunile FORTRAN nu sunt "case sensitive" (`CALL` şi `call` sunt echivalente).

EXERCIŢIUL 11

Folosind facilitatea Scilab de a apela rutine externe (paragraful 6.10), incorporaţi în Scilab, executaţi şi verificaţi în mediul Scilab rutina de calcul al integralei definite pe intervale nemărginite, `qagi`.

EXERCIŢIUL 12

Scrieţi un raport (referat de laborator) care să descrie modul de rezolvare a exerciţiilor de utilizare a unei rutine din biblioteca matematică `QUADPACK`.

Descrieţi scopul subrutinei `qagi`. Comentaţi dificultăţile apărute. Anexaţi codul sursă al programelor scrise, precum şi conţinutul fişierelor `Makefile` folosite.

Capitolul 8

Comunicaţia între sistemele Unix — Internet —

O caracteristică esenţială a sistemelor Unix este faptul că ele oferă implicit suportul software pentru comunicaţia între calculatoarele conectate. Evident că transmiterea de date între calculatoare depinde şi de caracteristicile hardware ale acestora, dar efortul enorm de standardizare şi normare (privind nivelul logic sau cel fizic: conectori, nivele de semnale, etc.) permite în prezent realizarea de reţele hibride în care sunt incluse cele mai diverse echipamente (calculatoare, servere de fişiere, echipamente de conectare, imprimante şi alte echipamente periferice), provenite de la producători diferiţi, pe baza modelului client/server. LMN este primul laborator din ţară care a realizat în 1992 o reţea hibridă de staţii grafice aplicând acest model.

Reţelele de calculatoare sunt structurate ierarhic pe nivele:

- **LAN** (Local Area Network) – reţeaua dintr-un laborator (cum este `lmn.pub.ro`), departament sau instituţie cu dimensiuni cuprinse între 5 m şi 30 km;
- **MAN** (Metropolitan Area Network) – reţeaua dintr-un oraş, obţinută prin conectarea mai multor LAN-uri, cu dimensiuni între 10 şi 100 km;
- **WAN** (Wide Area Network) – reţeaua de la nivelul unei ţări (cum este *RoEduNet*) sau chiar continent;
- **GAN** (Global Area Network) – reţeaua la nivel mondial (cum este *Internet*).

Cel mai simplu mod de conectare a două calculatoare este cel *punct-la-punct*, de exemplu folosind interfaţa serială (comuna majorităţii sistemelor de calcul) şi două fire conductoare prin care datele se transmit serializat (bit cu bit) cu viteze cuprinse între 4 kb/s până la 128 kb/s. Eventual, pentru a folosi liniile telefonice obişnuite trebuie utilizate două echipamente specializate numite *modemuri*.

Reţelele reale au topologii ale conexiunilor mai complicate, ca de exemplu:

- topologie **magistrală**, un fir pe care se pot conecta mai multe calculatoare
- topologie **inel**, la care conexiunea în buclă închisă elimină reflexiile;
- topologie **stea**, în care fiecare sistem este conectat direct la un echipament central numit "*hub*";
- topologie de **graf**, la care în fiecare nod se conectează un echipament de comunicaţie numit "*router*" şi care stabileşte căile prin care va circula mesajul de la sursă la destinaţie.

Pentru a realiza comunicaţia este nevoie de un canal de transmisie a datelor realizat fizic pe baza unui suport (mediu) de transmisie. Acesta poate fi:

- pereche de **fire torsadate** (*UTP= Unshielded Twisted Pair*) folosite la LAN-uri;
- **cablu coaxial**, de obicei cu impedanţa caracteristică de 50 Ω sau cablu TV, folosit la LAN-uri şi respectiv MAN-uri;
- cablu **optic**, folosit de la MAN-uri până la GAN-uri;
- unde **radio şi microunde**, folosite la MAN-uri;
- unde **infraroşii**, folosite la LAN-uri;
- linii de **telefon** (comutate sau dedicate), folosite la toate nivelele;
- telefonie **celulară** (analogică sau GSM), folosite la toate nivelele;
- **sateliţi** (geostaţionari sau cu orbită fixă *LEO – Low Earth Orbit*), folosiţi la WAN şi GAN.

În funcţie de suportul folosit, viteza de transmisie a datelor variază de la câţiva kb/s până la Gb/s. În LMN vitezele sunt de 10 Mb/s la reţelele bazate pe cablu coaxial din EB210 şi EB213 şi de 100 Mb/s la reţeaua cu UTP, în arhitectură stea, din EB212. Universitatea "Politehnica" din Bucureşti, ca nod central al reţelei Ministerului Educaţiei Naţionale – *RoEduNet*, are o reţea bazată pe cablu coaxial galben şi fibră optică între clădiri (cca 10 km), conexiunea cu exteriorul realizându-se prin radiomodem (2 Mb/s), antenă de satelit VSAT (1.5 Mb/s), cablu TV (5 Mb/s) şi prin modem special de conectare la linie telefonică dedicată (128 kb/s).

Istoria reţelelor de calculatoare Internet începe în perioada războiului rece, în 1969 când *ARPA* (*Defense Advanced Research Projects Agency*) a finanţat un proiect de conectare sigură a calculatoarelor în reţea pentru transmiterea de informaţii. Reţeaua *ARPAnet* lega în 1969 trei calculatoare ale unor universităţi californiene (UCLA, USC şi Stanford) cu Universitatea Utah. Reţeaua s-a extins progresiv acoperind tot teritoriul SUA: 8 sisteme în 1970, 15 sisteme în 1971, 35 sisteme în 1972, iar în 1973 a căpătat dimensiune transatlantică prin conexiunile cu Marea Britanie şi Norvegia.

Un rol important în dezvoltarea reţelelor de calculatoare l-a avut programul finanţat de *NSF* (*National Science Fondation*) care a permis realizarea, începând din 1979, a unei reţele dedicată cercetătorilor *NSFNet*. Începând din 1991 (prin legea numită *High Performance Computing – HPC Act*) dezvoltarea acestei reţele a condus la realizarea reţelei naţionale americane de cercetare şi educaţie (*National Research and Education Network – NREN*) la care sunt conectate cele mai mari supercalculatoare americane şi are o coloană vertebrală ("*backbone*") de mare viteză numită şi "*autostradă informatică*".

Creşterea numărului de calculatoare conectate a avut loc exponenţial (213 în 1981, 535 500 în 1991 şi peste 3 milioane în 1994) şi continuă şi în prezent.

În paralel s-au dezvoltat pe toate continentele reţele dedicate activităţilor nonprofit sau celor comerciale şi respectiv guvernamentale. Serviciile oferite utilizatorilor s-au diversificat continuu: poşta electronică **e-mail** (din 1977), agenţie de ştiri - **Usenet** (1979), hipertext **Web** (1992 la CERN în Elveţia) şi **browserul Mosaic** (1993 la NCSA în SUA).

Numele Internet s-a impus după 1983, când a fost realizat protocolul **IP** (*Internet Protocol*), care împreună cu **TCP** (*Transport Protocol*) permite conectarea în reţea nu numai a calculatoarelor simple ci şi a reţelelor locale de calculatoare (LAN). În acest fel Internet-ul este de fapt nu o reţea de calculatoare ci o reţea de reţele de calculatoare, fiecare având adresă IP proprie (pe 32 biţi, deci un total de cca 4 miliarde de adrese cuprinse între 0.0.0.0 şi 255.255.255.255). De exemplu adresele calculatoarelor din LMN sunt 141.85.34.xxx. În 1996 a apărut o versiune nouă a protocolului **IPv6**, care adaugă protecţii suplimentare de securitate şi care extinde lungimea adresei la 128 biţi astfel încât practic pe fiecare metru pătrat de pe suprafaţa locuită a planetei să se poată aloca mii de adrese.

Pentru a asigura flexibilitatea şi robusteţea necesară unei astfel de dezvoltări s-au făcut eforturi de stabilire a unor standarde şi norme internaţionale, în special de Organizaţia Internaţională de Standarde *ISO*. Cel mai cunoscut este modelul de transmisie a datelor *OSI* (*Open System Interconection*) prin care mesajul este prelucrat succesiv de 7 nivele (fiecare introducând eventual câte un antet special) şi prezentate în figura 8.1. În prezent comunicările se realizează folosind mai puţine nivele (4 în cazul Internet clasic).

Ideea centrală a transmisiei Internet (TCP/IP) este de a sparge mesajul în pachete de dimensiuni mai mici fiecare având câte un antet în care sunt codificate adresa sursă şi cea destinaţie ale mesajului. Pachetele unui mesaj pot în general circula pe rute diferite iar la destinatar ele se asamblează în ordinea corectă [78].

Un alt aspect este legat de accesul multiplu la un canal de transmisie (bandă radio sau magistrală). Primele cercetări au fost realizate în anul 1970 în cadrul proiectului *ALOHA* de comunicaţie prin radio între insulele Hawai. Rezultatele au fost preluate de Xerox PARC, unde în 1976 s-au conectat la un cablu de 1 km lungime peste 150 de staţii de lucru personale (la 3 Mb/s) într-o reţea locală numită *Ethernet*. Ulterior protocolul şi specificaţiile mecanice pentru conectori au fost cuprinse în standardul IEEE 802.3 cu două variante: *Ethernet* (10 Mb/s) şi *Fast Ethernet* (100 Mb/s), ultima apărută in 1995.

În tabelul 8.1 se prezintă soluţiile permise de acest standard.

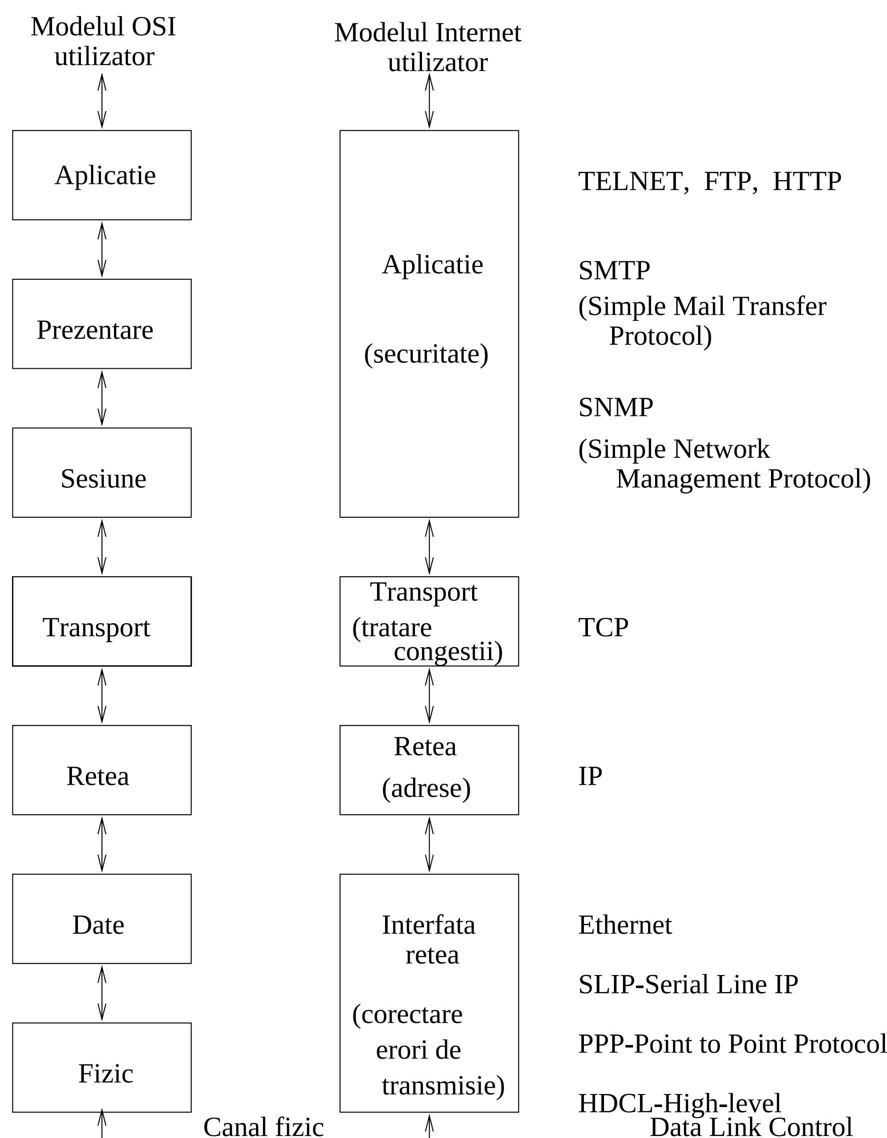


Fig. 8.1: Comparaţie între modelele OSI şi Internet

Orice reţea, indiferent de extensie, are câteva caracteristici definitorii, ca cele descrise în tabelul 8.2.

Dezvoltarea fără precedent a reţelei de calculatoare a impus pentru realizarea interoperabilităţii adoptarea unor norme tehnice comune. În acest sens, s-a creat în 1992 *Internet Society*, care are o procedură deschisă de redactare a proiectelor de norme, numită *RFC* (*Request For Comments*) [8]. Aici se menţin definiţiile diferitelor protocoale, cum sunt: IP, TCP, FTP, TELNET, SMTP, DNS (Domain Name Server, ce standardizează alocarea numelor de domenii si subdomenii) sau NFS (Network File Sistem, ce permite distribuirea în reţea a unui sistem de fişiere fără ca utilizatorul să fie preocupat de poziţionarea fizică a unui fişier).

În România primele încercări de conectare la Internet au avut loc la *Catedra de Calculatoare* din UPB în iulie 1990 (transmisie de poştă electronică pe linie comutată cu Uni-

Tabelul 8.1: Standardul Ethernet IEEE 802.3

Nume	**Cablu**	**Lungime max. segment**	**Nr. max. noduri / segment**	**Avantaje**
10 Base 5	cablu coaxial gros (galben)	500 m	100	Backbone
10 Base 2	cablu coaxial subţire (negru)	200 m	30	Ieftin
10 Base T	Torsadat	100 m	1024	Uşor de menţinut
10 Base F	cablu optic	2000 m	1024	Între clădiri
100 Base T4	Torsadat	100 m	1024	UTP 3
100 Base-TX	Torsadat	100 m	1024	UTP 5
100 Base-F	fibre optice	2500 m	1024	Lungime

Tabelul 8.2: Caracteristicile principale ale unei reţele

Caracteristica	**Posibilităţi**
Serviciu	cu sau fără conexiune fixă
Protocol	IP, IPX
Adresare	plată (802) sau ierarhică (IP)
Multitasking	prezent sau absent
Mărime pachete	scurte sau lungi
Calitate	sub control sau nu
Corector erori	da sau nu
Transmisie	cu controlul vitezei sau nu
Controlul congestiei	cu diverse tehnici
Securitate	cu sau fără encriptare
Contabilitate	după timp, cantitate de informaţie, nr. pachete sau fără

versitatea Tehnică din Darmstadt – THD).În februarie 1993 s-a dat în folosinţă reţeaua locală a UPB, una din cele mai mari din Europa, care în iunie 1993 a fost conectată prin linie telefonică dedicată la 9.6 kb/s cu Berlin. Domeniul `.ro` a apărut pe harta Internet în anul 1993 prin eforturile *Institutului Central de Informatică* [43], [60].

În 1994 reţeaua UPB a deschis accesul la Internet şi altor universităţi din ţară. În prezent reţeaua *RoEduNet* are arhitectură stea cu nodul central în UPB şi alţi cinci Internet Service Providers în Craiova, Timişoara, Cluj, Tg.Mureş şi Iaşi, oferind servicii pentru practic întreaga comunitate universitară din ţară.

Dintre aceste servicii, cel care a contribuit cel mai mult la impactul Internet-ului (până la a se confunda cu acesta) este cel de *Web* [64]. Acest serviciu se bazează pe conceptul de hipertext şi dă posibilitatea de a vizualiza documente răspândite în întrega lume cu subiectele cele mai diverse, dar "legate" logic între ele prin "ancore" (cuvinte sensibile, marcate în document prin culoare sau subliniere şi prin a căror indicare se realizează automat transferul la documentul cu care acest cuvânt este legat). Hipertextele sunt scrise într-un limbaj specializat numit *HTML* (*Hypertext Mark-up Language*) [68].

În acest fel se realizează o bibliotecă electronică cu o topologie asemănătoare unei pânze de păianjen (Web), distribuită pe întreg globul (*WWW = World Wide Web*). Această bibliotecă sporeşte continuu, deoarece orice utilizator are posibilitatea să afişeze informaţie în acest ciberspaţiu. Sistemele pe care sunt memorate documentele se numesc *Web-servere*, iar programul client care permite vizualizarea lor de la distanţă se numeşte *browser* [76]. Operaţiile pe care utilizatorul Web le poate realiza sunt:

- navigaţia
- explorarea
- producţia (folosind următoarele tehnici: editare *off-line*, *WYSIWYG* sau prin traducere din alt limbaj)

Documentele pot conţine nu numai text ci şi imagini (*GIF* şi *JPEG*), iar în prezent chiar elemente multimedia: sunete, animaţii şi secvenţe video, realizându-se ceea ce se numeşte "*hipermedia*". Mai mult, nu toate "documentele" sunt statice, ci unele permit interacţiunea cu utilizatorul (descrisă prin intermediul unor limbaje de programare specializate, cum sunt: *CGI, Pearl, Java, JavaScript*, etc.) [66].

Normele şi standardele privind Web-ul sunt menţinute de *W3C – World Wide Web Consortium* [28], înfiinţat în acest scop în 1994.

Perspectivele deschise de reţeaua de calculatoare Internet şi de abordarea client-server au făcut ca filozofia privind utilizarea calculatoarelor să fie total modificată în ultimii ani, iar în perspectivă este aşteptată o adevărată revoluţie cu mare impact social [41]. Nu întâmplător firma Sun şi-a ales deviza "*The Network is the Computer*".

Obiectivul capitolului constă în însuşirea deprinderilor necesare folosirii principalelor servicii Internet, realizate de programe Unix care permit comunicarea directă cu alţi

utilizatori, conectarea la un calculator aflat la distanţă şi respectiv tranferul de fişiere de la / spre un calculator aflat la distanţă şi conectat la Internet:

- Comunicare directă cu alţi utilizatori:
  - `elm` — primire şi transmitere de mesaje (poştă electronică), care a fost tratat deja în capitolul 3;
  - `write` — comunicare unidirecţională cu alţi utilizatori;
  - `talk` — comunicare interactivă (bidirecţională) cu alţi utilizatori;
- Conectare (login) pe alte calculatoare:
  - `telnet`
- Transfer de fişiere de pe alte calculatoare:
  - `ftp`
- Alte comenzi utile
  - `mesg`
  - `uuencode, uudecode`
  - `gzip, compress`
  - `tar`
  - `ping`
  - `finger`
- World Wide Web
  - elemente de bază WWW
  - limbajul HTML
  - coduri HTML

8.1 telnet

Acest utilitar permite conectarea la o altă maşină, folosind un terminal alfa–numeric.

Pentru a testa această comandă, procedaţi astfel:

- Tastaţi:

  `telnet` *maşină*

  unde *maşină* este "numele" unei maşini pe care aveţi cont, sau pe care există un cont cu acces public (de exemplu, numele altei maşini decât cea pe care sunteţi conectat în laborator). Va urma o procedură standard de login pe acea maşină.

- Din acest moment, sunteţi conectat la acea maşină şi puteţi da comenzi ca şi cum v-aţi afla în faţa unui terminal al acelei maşini (cu observaţia ca, lucrând pe un terminal alfanumeric, comenzile care implică deschiderea de ferestre grafice sunt fie inaccesibile, fie necesită proceduri speciale).

- Pentru ieşire, tastaţi

  `logout <↵ >`

  sau

  `exit <↵ >`.

  Veţi reveni la promptul maşinii locale.

- Comanda `telnet` fără parametri are ca efect intrarea în programul `telnet`, afişarea promptului specific (`telnet >`) şi aşteptarea unei comenzi. Puteţi folosi comenzile:

  `open` *maşină*: conectarea la *maşină*

  `close`: întreruperea legăturii la *maşină*

  `quit`: ieşirea din `telnet`

8.2 write

Utilitarul Unix `write` permite comunicarea unidirecţională cu alţi utilizatori conectaţi pe aceeaşi maşină.

Pentru a testa modul de funcţionare a acestei comenzi, procedaţi astfel:

- Verificaţi că puteţi primi mesaje, cu comanda `mesg <↵ >`. Dacă rezultatul acestei comenzi este mesajul `Is n`, setaţi `mesg y`.

- Stabiliţi un "partener de discuţie" şi conectaţi-vă la maşina lui (folosind, de exemplu, `ssh`).

- Verificaţi dacă utilizatorul este conectat:

  `finger` *nume*

  sau

  `who`

- Dacă utilizatorul are deschis cel puţin un terminal pe maşina respectivă, rezultatul este similar cu:

```
irina    ttyp0   Oct 30 10:48   (:0.0)
irina    ttyp1   Oct 30 10:06   (:0.0)
irina    ttyp2   Oct 30 10:06   (:0.0)
```

  "Numele" terminalelor pe care lucrează utilizatorul `irina` este `ttyp.`.

- Introduceţi comanda:

  `write` *nume terminal*

  (*terminal* poate lipsi), apoi introduceţi textul mesajului, de exemplu:

  `test write <↵ >`

- În acest moment puteţi:

  – încheia transmiterea mesajului: Ctrl-D sau Ctrl-C
  – continua cu scrierea altei linii, terminate cu `<↵ >`.

Avantajul utilizării `write` în locul poştei electronice (e-mail) este acela că destinatarul primeşte imediat mesajul, care devine vizibil pe propriul terminal (metoda este sincronă şi nu asincronă, nefiind nevoie ca destinatarul să facă vreo acţiune pentru a primi mesajul). Această metodă este mai "agresivă" decât e-mail şi se recomandă să o folosiţi doar în cazurile în care sunteţi convins că destinatarul nu este deranjat. Imaginaţi-vă ce se întâmplă dacă destinatarul primeşte 10 mesaje simultan, într-un moment în care este preocupat de rezolvarea unei probleme ce nu suportă amânare.

8.3 talk

Utilitarul `talk` permite comunicarea interactivă (bidirecţională) cu alţi utilizatori conectaţi pe aceeaşi maşină sau pe maşini diferite.

Pentru a iniţia o sesiune `talk` procedaţi astfel:

- Verificaţi că puteţi primi mesaje, cu comanda `mesg <↵ >`. Dacă rezultatul acestei comenzi este mesajul `Is n`, setaţi `mesg y`.

- Stabiliţi un "partener de discuţie" şi verificaţi la ce maşină este conectat, prin comanda:

  `rwho` (comandă non–standard, disponibilă în laboratorul EB–212)

  sau

  `finger` *nume*`@`*maşină*

- Introduceţi comanda:

  `talk` *nume*`@`*maşină terminal*

  (*terminal* poate lipsi). Veţi vedea pe ecran mesajul:

  `[Waiting for your party to respond]`

  Dacă corespondentul nu răspunde, va apărea de câteva ori mesajul:

  `[Ringing your party again]`

  după care se renunţă.

- În momentul în care corespondentul a acceptat să converseze, ecranul se va împărţi în două, în partea de sus apărând textul tastat de Dvs, în partea de jos textul introdus de cealaltă persoană.

- Când doriţi să încheiaţi conversaţia, tastaţi:

  `bye`

  `Ctrl-C`

Pentru a răspunde la o "invitaţie" `talk` procedaţi astfel:

- Pe ecran va apărea un mesaj de tipul:

  ```
  Message from Talk_Daemon@lmnsun at 12:52 ...
  talk:  connection requested by irina@lmnsun.lmn.pub.ro.
  talk:  respond with:  talk irina@lmnsun.lmn.pub.ro
  ```

- Răspundeţi cu comanda indicată pe ultima linie a mesajului:

  `talk irina@lmnsun.lmn.pub.ro`

  (Atenţie, `irina@lmnsun.lmn.pub.ro` se înlocuieşte cu adresa corespondentului Dvs.)

- Puteţi lua iniţiativa încheierii conversaţiei (deşi nu este politicos, deoarece nu aţi fost cea/cel care a lansat-o!). Pentru aceasta tastaţi:

  `bye`

  `Ctrl-C`

8.4 ftp

Utilitarul `ftp` (File Transfer Protocol) permite conectarea la o altă maşină cu scopul de a transfera fişiere spre / de la acea maşină.

- Introduceţi comanda:

  ```
  ftp ftp.lmn.pub.ro
  ```

  Observaţie: `ftp.lmn.pub.ro` poate fi înlocuit cu orice altă adresă cu facilităţi de `ftp`.

- Veţi vedea un mesaj similar cu:

  ```
  220 lmnroute FTP server (Version wu-2.4(11) Mon Jul 10 19:14:15 GMT+0300
  1995) ready.
  Name (ftp.lmn.pub.ro:irina):
  ```

  Introduceţi numele de utilizator `anonymous` şi ca parolă, adresa Dvs. de poştă electronică (*nume*`@lmn.pub.ro`).

  De la caz la caz, numele de login dedicat accesului public poate fi `anonymous`, `guest`, sau alt nume. În general, vi se solicită o parolă, dar există locaţii ftp unde nu este nevoie de parolă.

- Din acest moment aveţi la dispoziţie o serie de comenzi pentru listarea conţinutului directoarelor, deplasarea în alte directoare, transferul de fişiere, etc. O listă selectivă a acestor comenzi este prezentată mai jos.

- Ieşiţi din `ftp` cu una din comenzile: `bye` sau `quit`

Listă selectivă de comenzi ftp

`ascii`	-	setează tipul de transfer de fişiere ca “ASCII”
`binary`	-	setează tipul de transfer de fişiere ca “binar”
`cd` *director*	-	schimbarea directorului curent, pe maşina de la distanţă (“remote”)
`lcd` *director*	-	schimbarea directorului curent, pe maşina locală (“local”)
`dir` *director* sau `ls` *director*	-	afişare conţinut director pe maşina de la distanţă
`put` *fis_local*	-	copiază fişierul local pe maşina de la distanţă
`get` *fis_dist*	-	copiază fişierul de pe maşina de la distanţă pe maşina locală
`mput` *fis_local*	-	asemănător cu `put` dar pentru mai multe fişiere. De exemplu: `mput *.c`
`mget` *fis_dist*	-	asemănător cu `get` dar pentru mai multe fişiere
`hash`	-	afişează câte un caracter `#` pentru fiecare bloc de date transferat; permite astfel vizualizarea evoluţiei transferului
`open`	-	iniţierea unei sesiuni `ftp`, de la promptul ftp

`close`	-	încheie sesiunea cu maşina de la distanţă, fără să părăsească `ftp`
`user`	-	setează numele de utilizator ftp pe maşina de la distanţă (util în cazul în care aţi greşit numele de utilizator sau parola)
`bye`	-	încheie sesiunea `ftp`

8.5 Alte comenzi utile, legate de transferul de fişiere şi de comunicaţie

- `rlogin` *maşină* sau `ssh` *maşină*

  Permite conectarea la o altă maşină, folosind un terminal alfa–numeric. Se recomandă utilizarea comenzii `ssh` care este mai sigură (encriptează parola).

- `mesg`

  setarea acceptării / neacceptării mesajelor din exterior

- `gzip`, `gunzip`:

  compactarea/decompactarea fişierelor. Fişierele compactate rezultate au extensia `.gz`

- `compress`, `uncompress`:

  compactarea/decompactarea fişierelor. Fişierele compactate rezultate au extensia `.Z`. Comanda `compress` este mai puţin eficientă (din punctul de vedere al reducerii dimensiunii fişierului) decât `gzip`.

- `uuencode`, `uudecode`:

  transformarea fişierelor binare în fişiere ASCII şi respectiv revenirea la forma binară a fişierelor Se folosesc pentru transmiterea unor fişiere pe "căi" care nu acceptă decât formatul text (ASCII), cum ar fi poşta electronică.

- `tar`:

  arhivare fişiere, directoare, etc. Cu ajutorul acestei comenzi, un sistem structurat format din mai multe fişiere este arhivat într-un singur fişier care include toate informaţiile pentru a putea reconstitui structura iniţială la dezarhivare.

  Exemple:

  – `tar cvf arhiva.tar test*`

    Crează fişierul `arhiva.tar`, care conţine arhiva tuturor fişierelor şi directoarelor al căror nume începe cu `test`, plasate în directorul curent.

    Parametrii au semnificaţiile:

    `c` ("**c**reate") – crearea unei noi arhive.

    `v` ("**v**erbose") – determină afişarea numelor fişierelor incluse, pe măsură ce ele sunt incluse în arhivă.

`f` ("file") – indică faptul că următorul parametru este numele fişierului–arhivă.

– `tar tvf arhiva.tar`

  Afişează ("list") conţinutul arhivei plasată în fişierul `arhiva.tar`.

– `tar xvf arhiva.tar`

  Extrage ("extract") fişierele cuprinse în arhiva `arhiva.tar`, reconstituind structura de fişiere şi directoare.

8.6 Elemente de bază WWW

Pentru a accesa serviciile *Internet–World Wide Web* (*WWW*) este necesar un "browser" — un program care citeşte şi afişează pe ecranul calculatorului Dvs. informaţiile aflate pe alte calculatoare conectate la Internet.

Browserele cele mai folosite sunt *Netscape* (figura 8.2) şi Lynx (cel din urmă, dedicat terminalelor alfa–numerice, deci neoferind facilităţi grafice).

Orice informaţie pe Internet este accesibilă prin "adresa" sa, numită *URL* (*Uniform Resource Locator*).

Un URL este compus din trei părţi:

protocol://server/cale

protocol indică browser–ului cum să interpreteze informaţia găsită pe maşina ***server*** în fişierul indicat prin ***cale***. Dacă parametrul ***cale*** reprezintă un director şi nu un fişier, atunci browserul va prelua în mod implicit fişierul `Index.html` plasat în directorul indicat prin ***cale***.

protocol poate fi:

`http` — hiperdocument în format HTML

`gopher` — pentru lansarea unei sesiuni `gopher`

`ftp` — pentru lansarea unei sesiuni `ftp`

`file` — pentru accesarea unui fişier care poate fi în format HTML sau nu.

Exemplu de URL (adresa paginii **WWW** şi a celei **ftp** a Laboratorului de Metode Numerice):

`http://www.lmn.pub.ro/`

`http://archive.lmn.pub.ro/`

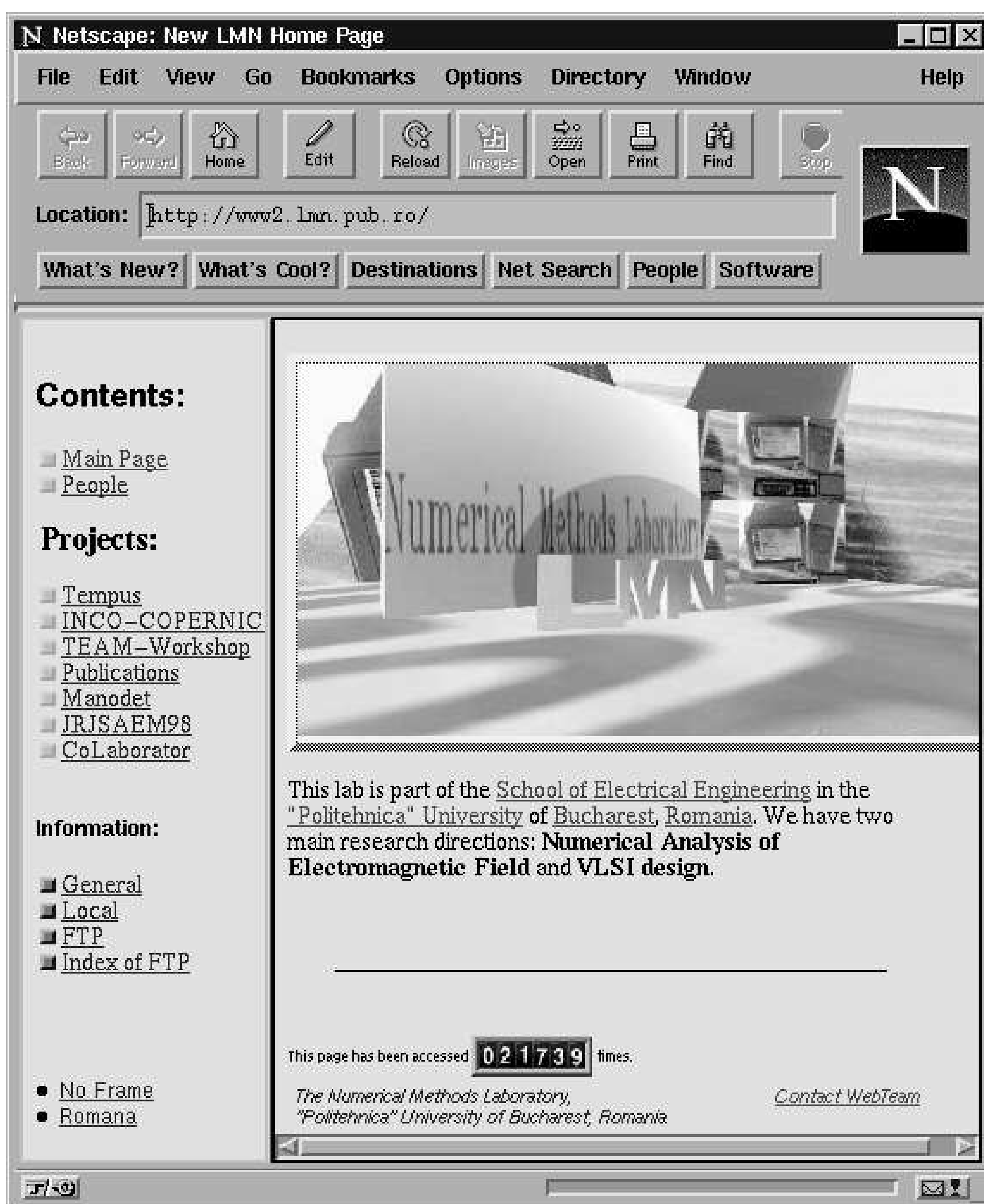


Fig. 8.2: Ecranul browser-ului Netscape

8.7 Structura limbajului HTML

Documentele pe Internet au un format special, numit *HTML* (*HyperText Markup Language* = limbaj bazat pe marcaje pentru hipertext). Noţiunea de hipertext desemnează o structură a informaţiei care nu este liniară, ca de exemplu într-un film, ci arborescentă, semănând cu o carte pe care un utilizator o poate "frunzări" după voie şi chiar este ajutat să o facă în mod eficient prin index sau alte metode de trimitere.

Documentele hipertext conţin coduri speciale care sunt interpretate de browser, coduri referitoare la:

- formatarea textului
- includerea în text a unor imagini
- preluarea unor informaţii de pe Internet,

  ...., etc.

Trebuie menţionat că:

- orice fişier ASCII este afişabil (vizibil) din orice browser; nu este deci neapărat nevoie să ştiţi HTML pentru a afişa un document pe Web;
- totuşi, un document formatat HTML arată mai atractiv. În plus, în el se pot include legături ("links") la alte pagini (a se citi "fişiere"!) plasate pe maşina Dvs. sau pe orice altă maşină accesibilă din lume.

Un document formatat HTML se numeşte în jargonul curent "pagină Web" (*Web page*, WWW page), iar mulţimea paginilor unei instituţii se numeşte *Web site*. Fişierul care conţine documentul se numeşte "fişier sursă" şi este un fişier text, având extensia standard `.hmtl` sau `.htm`.

Pagina de "bază" a unei instituţii sau a unui utilizator este numită adesea "home page". Pentru cazul unui utilizator, este de obicei plasată în directorul `~/public_html/` şi are numele `Index.html`.

Conceptual, limbajul HTML se aseamănă cu LaTeX, amândouă satisfăcând prescripţiile *SGML* (*Standard General Markup Language*). Diferă doar sintaxa, dar conceptele care stau la baza celor două limbaje sunt similare. Din acest motiv, traducerea între aceste limbaje este relativ uşoară.

8.8 Coduri de formatare HTML

Codurile de formatare (în limba engleză: "tags") servesc la marcarea unei "indicaţii de punere în pagină" de către browser a documentului HTML.

Un tag este structurat astfel: semnul <, urmat de un cod (numit "directivă"), urmat de semnul >:

```
<directivă>
```

Tag–urile sunt de obicei în perechi, de exemplu:

```
<H1> Text </H1>
```

(un titlu de nivel 1, definit prin directiva H1).

Observaţi similitudinea cu:

```
\begin{document} Text \end{document}
```

din LaTeX.

Un document HTML are structura minimală:

```
<HTML>

<HEAD>
<TITLE>   Titlul paginii </TITLE>
</HEAD>

<BODY>
.  .  .
</BODY>

</HTML>
```

8.8.1 Principalele perechi de coduri de formatare HTML

- Heading pentru întregul document

  ```
  <HEAD> ...  </HEAD>
  ```

- Titlu document

  ```
  <TITLE> ...  </TITLE>
  ```

- Titluri secţiuni

  ```
  <H1> ...  </H1>
       ⋮
  <H6> ...  </H6>
  ```

- Formatare text

  `<I> Italic </I>` — *Italic*

  `<B> Bold </B>` — **Bold**

  `<U> Subliniat </U>` — Subliniat

- Paragrafe

  `<P> ... </P>`

  Observaţie: `</P>` poate lipsi.

- Linie nouă

  `<BR>`

- Listă nenumerotată

```
<UL>
     <LI> ... </LI>
     <LI> ... </LI>
          ⋮
</UL>
```

  Observaţie: `</LI>` poate lipsi.

- Listă numerotată

```
<OL>
     <LI> ... </LI>
     <LI> ... </LI>
          ⋮
</OL>
```

  Observaţie: `</LI>` poate lipsi.

- Linie orizontală

  `<HR>`

- Includere imagine

  <IMG SRC = *URL* ALIGN = "*aliniere*" ALT = "*alternativă*">

  Exemplu:

  `<IMG SRC = http://www.lmn.pub.ro/graphics/im1.gif>`

  (Atenţie! Numele fişierului –`/graphics/im1.gif`– este doar un exemplu!)

  Opţiunile `ALIGN` şi `ALT` pot lipsi.

  Parametrul *aliniere* poate avea valorile: `top`, `middle`, `bottom` şi se referă la alinierea figurii faţă de textul care urmează.

Opţiunea ALT este folosită de către browser-ele care nu permit afişări grafice. Textul *alternativă* se va afişa pe ecran, în locul unde ar fi trebuit să apară figura.

Exemplu:

```
<IMG SRC = "poza.gif" ALIGN ="middle" ALT = "GIF image"> O imagine
```

- Includerea unei legături la un alt document Internet

  <A HREF = "*URL*"> *text* </A>

  Perechea de tag–uri este acum <A> şi </A>. Textul *text* este afişat diferit faţă de restul textului; forma cursorului se va modifica atunci când acesta se mută pe *text*.

  Exemplu:

```
<A HREF = "alt_inf.html"> Apasati aici pentru alte informatii </A>
```

8.9 Un exemplu de document HTML

Iată un exemplu de utilizare a codurilor prezentate în paragraful anterior:

```
<HTML>
<HEAD>
<TITLE> Pagina de test </TITLE>
</HEAD>

<BODY>

<H1> Un titlu mare </H1>
<H3> Un titlu mai mic </H3>

Acum urmeaza o lista:
<UL>
<LI> mere
<LI> pere
<LI> portocale
</UL>

Si inca o lista
<OL>
<LI> Primul
<LI> Al doilea
<LI> Al treilea
</OL>

O linie orizontala:
<HR>

Gata!
</BODY>
```

... şi rezultatul – ceea ce se vede atunci când folosiţi browser-ul Netscape pentru a vizualiza documentul:



8.10 Exerciţii

EXERCIŢIUL 1 — **"Navigare"**

Timp de 15 minute, navigaţi pe Internet folosind Netscape. Veţi constata că minutele trec foarte repede în cursul acestei activităţi – uitaţi-vă din când în când la ceas!

Vă recomandăm să atingeţi câteva puncte: pagina LMN `http://www.lmn.pub.ro`, pagina UPB `http://www.pub.ro` şi pagina Netscape `http://www.netscape.com`.

EXERCIŢIUL 2 — **"Explorare"**

Folosind programele de căutare a informaţiilor plasate în Internet, (o listă este disponibilă la `http://www.lmn.pub.ro/wwwsearch.html`) realizaţi o documentare pe Internet privitoare la biblioteci de subrutine matematice.

Începeţi să vă construiţi propriul "carnet de adrese" Internet: salvaţi adresele Internet utile folosind butonul **Bookmarks** → **Add bookmark**.

Folosiţi editorul dedicat (**Window** → **Bookmarks**) pentru a vă organiza adresele Internet. Creaţi foldere pe domenii (**Item** → **Insert Folder**), în care mutaţi (apăsaţi butonul din stânga şi deplasaţi mouse-ul) adresele corespunzătoare.

Figura 8.3 prezintă aspectul ferestrei Netscape de gestionare a adreselor Internet.

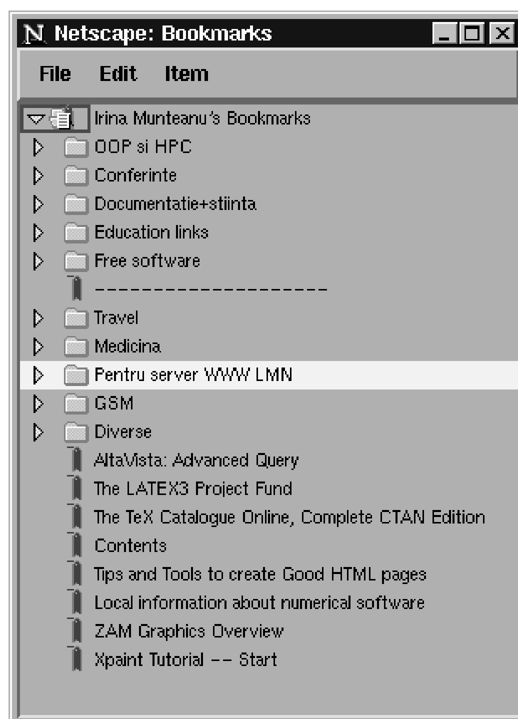


Fig. 8.3: Fereastra "Bookmarks" din Netscape

EXERCIŢIUL 3 — **"Creare de informaţie electronică"**

Scopul acestui exerciţiu este de a vă crea propria pagină WWW, pornind de la un model.

1. Creaţi directorul `public_html` în directorul Dvs. $HOME.
2. Accesaţi cu Netscape adresa: `http://www.lmn.pub.ro/~irina`.
3. Salvaţi documentul ca "sursă HTML" în fişierul `~/public_html/Index.html`
4. Accesaţi cu Netscape fişierul nou creat, selectând adresa:

   `http://www2.lmn.pub.ro/~`*nume*

   unde *nume* este numele Dvs. de utilizator.
5. Deschideţi o nouă fereastră şi editaţi cu `vi` sau cu `emacs` fişierul `Index.html`
6. Identificaţi principalele părţi ale documentului şi tag–urile folosite (comparaţi cu lista de tag–uri din paragrafele următoare).
7. Modificaţi fişierul astfel încât să conţină propriile Dvs. informaţii. Salvaţi fişierul fără să ieşiţi din editor.
8. Verificaţi rezultatul modificărilor folosind comanda "Reload".
9. Dacă rezultatul este satisfăcător, închideţi fişierul Index.html.

EXERCIŢIUL 4

Deschideţi pagina Dvs. WWW folosind editorul Netscape (butonul "Edit").

Familiarizaţi-vă cu facilităţile acestui editor. Experimentaţi diferite fonturi, linii de separare, includere de figuri, etc.

Includeţi în această pagină ancore la referatele de laborator realizate până acum în format ASCII sau PostScript.

EXERCIŢIUL 5

Realizaţi o pagină cu preferinţe (**TOP 10**), în care descrieţi de exemplu cele mai interesante 10 instrumente Unix, (eventual la fiecare cele mai utile 10 comenzi), cele mai interesante 10 pagini Web pe care le-aţi descoperit, cele mai interesante 10 cărţi, etc.

În pagina Dvs. home includeţi o ancoră la pagina TOP 10. În acest fel, pagina de Web va reflecta personalitatea Dvs.!

EXERCIŢIUL 6

Folosiţi programul `latex2html` pentru a traduce automat în format HTML referatul de laborator referitor la mediul de programare C în Unix. În pagina proprie WWW, adăugaţi o ancoră la acest document HTML şi verificaţi că el este vizibil din Netscape.

EXERCIŢIUL 7

Adăugaţi adresa paginii Dvs. WWW în fişierul `logo` inclus automat în orice mesaj electronic.

Trimiteţi un mesaj instructorului, în care îi comunicaţi finalizarea **ultimului exerciţiu** al acestei cărţi!

Anexa A

Teste finale de laborator

Testul 1 de verificare a cunoştinţelor — Tutorial Unix

1. Creaţi (temporar, pentru durata acestui test) un director numit `dir1` în directorul Dvs. $HOME. În acest director veţi lucra pe parcursul testului.
2. Transferaţi în directorul `dir1` fişierul `test_unix` aflat pe maşina `ciuc` în directorul `/tmp`.
3. Copiaţi fişierul `test_unix` în fişierul `test.txt`
4. Editaţi fişierul `test.txt`:
   - înlocuiţi toate apariţiile cuvântului `test` cu cuvântul `text`.
   - înlocuiţi orice succesiune de două spaţii (blancuri) cu un singur blanc
   - ştergeţi toate liniile care conţin cuvintele `de sters`
   - salvaţi şi ieşiţi din editor.
5. Copiaţi fişierul `test.txt` în fişierul `test1.txt`
6. Listaţi conţinutul directorului `dir1`, redirectând ieşirea (afişarea) în fişierul `dir`
7. Trimiteţi prin poştă electronică două mesaje pe adresa `irina`:
   - unul cuprinzând fişierul `test1.txt`
   - unul cuprinzând fişierul `dir`
8. În mod normal, 10 minute vă sunt suficiente pentru a termina acest test. **Aveţi la dispoziţie 40 minute.**

Succes !

Testul 2 de verificare a cunoştinţelor — Tutorial Unix

1. Creaţi (temporar, pentru durata acestui test) un director numit `dir1` în directorul Dvs. $HOME. În acest director veţi lucra pe parcursul testului.
2. Transferaţi în directorul `dir1` fişierul `test_unix` aflat pe maşina `ciuc` în directorul `/tmp`.
3. Redenumiţi fişierul `test_unix` ca `test.tex`
4. Editaţi fişierul `test.tex` pentru a fi un document LaTeX:
   - folosiţi stilul `article`
   - adăugaţi titlul `Test de verificare a cunoştinţelor`
   - adăugaţi numele Dvs. (ca autor) şi data de astăzi
   - adăugaţi la sfârşit formula numerotată:

     $$y = \sum_{k=1}^{n} c_k x_k$$

   - salvaţi şi ieşiţi din editor.
5. Compilaţi şi eliminaţi eventualele erori.
6. Trimiteţi prin poştă electronică un mesaj pe adresa `irina`, cuprinzând fişierul `test.tex`
7. În mod normal, 10 minute vă sunt suficiente pentru a termina acest test. **Aveţi la dispoziţie 40 minute.**

Succes !

Testul 3 de verificare a cunoştinţelor — Tutorial Unix

1. Creaţi (temporar, pentru durata acestui test) un director numit `dir1` în directorul Dvs. $HOME. În acest director veţi lucra pe parcursul testului.

2. Scrieţi un macro Scilab care:

   pentru i = 1, 3

   preia de la consolă două matrice (sau vectori, sau scalari ...)

   efectuează produsul lor.

3. Rulaţi programul şi verificaţi corectitudinea

4. Transferaţi în directorul `dir1` fişierul `test_unix` aflat pe maşina `ciuc` în directorul `/tmp`.

5. Redenumiţi fişierul `test_unix` ca `test.tst`

6. Listaţi conţinutul directorului `dir1`, redirectând ieşirea (afişarea) în fişierul `dir`

7. Trimiteţi prin poştă electronică două mesaje pe adresa `irina`:

   - unul cuprinzând fişierul în care aţi scris programul Scilab
   - unul cuprinzând fişierul `dir`

8. În mod normal, 10 minute vă sunt suficiente pentru a termina acest test. **Aveţi la dispoziţie 40 minute.**

Succes !

Testul 4 de verificare a cunoştinţelor — Tutorial Unix

1. Creaţi (temporar, pentru durata acestui test) un director numit `dir1` în directorul Dvs. $HOME. În acest director veţi lucra pe parcursul testului.

2. Folosind `gnuplot`, reprezentaţi pe acelaşi grafic funcţiile $\sin(x)$ şi $\cos(x^2)$, cu următoarele setări:

   - eticheta axei Ox: `x`
   - eticheta axei Oy: `f(x)`
   - graficul pentru funcţia sin: puncte şi linii, titlu `sinus`
   - graficul pentru funcţia cos: puncte titlu `cosinus`

3. Salvaţi graficul în format Encapsulated PostScript (eps) în fişierul `grafic.eps`

4. Transferaţi în directorul `dir1` fişierul `test_unix` aflat pe maşina `ciuc` în directorul `/tmp`.

5. Redenumiţi fişierul `test_unix` ca `test.nou`

6. Copiaţi fişierul `test.txt` în fişierul `test1.txt`

7. Listaţi conţinutul directorului `dir1`, redirectând ieşirea (afişarea) în fişierul `dir`

8. Trimiteţi prin poştă electronică două mesaje pe adresa `irina`:

   - unul cuprinzând fişierul `grafic.eps`
   - unul cuprinzând fişierul `dir`

9. În mod normal, 10 minute vă sunt suficiente pentru a termina acest test. **Aveţi la dispoziţie 40 minute.**

Succes !

Testul 5 de verificare a cunoştinţelor — Tutorial Unix

1. Pe parcursul acestui test veţi lucra în directorul `public_html` (aflat în directorul Dvs. $HOME). Dacă directorul `public_html` nu există, creaţi-l.

2. Transferaţi în directorul `dir1` fişierul `test_unix` aflat pe maşina `ciuc` în directorul `/tmp`.

3. Redenumiţi fişierul `test_unix` ca `test.html`

4. Editaţi fişierul `test.html` astfel încât să devină un fişier în limbaj HTML:
   - adăugaţi marcajele (tag–uri) absolut necesare într-un document HTML
   - adăugaţi titlul `Test de verificare a cunoştinţelor`
   - adăugaţi numele Dvs. la începutul textului, ca heading de nivel 1.
   - adăugaţi un marcaj (tag) pentru trecerea la un nou paragraf, astfel încât textul pe ecran să cuprindă doar două paragrafe.
   - adăugaţi o linie orizontală la sfârşitul textului
   - salvaţi şi ieşiţi din editor.

5. Verificaţi cu NetScape că rezultatul modificărilor este cel aşteptat.

6. Listaţi conţinutul directorului curent (`public_html`), redirectând ieşirea (afişarea) în fişierul `dir`

7. Trimiteţi prin poştă electronică două mesaje pe adresa `irina`:
   - unul cuprinzând fişierul `test.html`
   - unul cuprinzând fişierul `dir`

8. Ştergeţi fişierele `test.html` şi `dir`.

9. În mod normal, 10 minute vă sunt suficiente pentru a termina acest test. **Aveţi la dispoziţie 40 minute.**

Succes !

Anexa B

Regulamentul Laboratorului de Metode Numerice

Instrucţiuni de respectat la folosirea reţelei de calculatoare locale din Laboratorul de Metode Numerice (`lmn.pub.ro`)

Oricine intră în laboratorul de metode numerice şi are deschis un cont activ pe reţeaua locală de calculatoare trebuie să cunoască următoarele reguli.

1. La fiecare intrare in sistem (login time) veţi primi diferite mesaje de care trebuie sa ţineti cont. Acestea sunt scrise de către administratorul de sistem şi au forma:

   ```
   ###########################################
   Incercati sa stergeti din fisierele care nu
   mai sunt necesare
   ###########################################
   ```

2. Încercati să vă ţineţi cât mai actuale datele personale cu ajutorul comenzii chfn şi a fisierului .plan

3. Accesul cu diskete este total interzis. Orice problemă care cere introducerea unei diskete din exteriorul laboratorului va fi soluţionată numai de administrator.

4. Accesul fizic la calculatoare se va face numai prin intermediul tastaturii şi a mouse-ului. Nu acţionati nici un alt buton. Nu deconectaţi/conectaţi cabluri.

5. Întrucât aveţi şi acces direct la reţeaua Internet va trebui sa fiţi un bun locuitor al acesteia. Este interzis accesul în orice calculator din lume dacă nu aveţi un cont personal. Orice încercare de pătrundere frauduloasă în orice sistem sau de producere a oricăror daune asupra datelor din alte sisteme se va pedepsi foarte drastic.

6. Parola este personală şi este interzisă folosirea contului de către orice altă persoană în afară de posesorul de drept al acestuia. Nu incredintaţi parola nimănui altcuiva. Schimbaţi parola cel puţin lunar cu ajutorul comenzii yppasswd. Alegeţi o parolă bună, în primul rând foarte diferită de numele de cont. Aceasta trebuie să aibă 6-8 caractere, litere şi cifre.

7. Este interzisă înlesnirea accesului altor persoane în contul propriu via .rhosts.

8. Încercaţi să vă rezolvaţi singuri problemele prin metode soft care nu cer drepturi de root. Când sunteţi siguri că nu se poate, scrieţi o hârtie cu datele amănunţite ale problemei, cum aţi încercat s-o rezolvaţi şi numele d-vstră de cont. Mai sigur prin mail, la adresa:

   `help@lmn.pub.ro`

9. Orice resursă costă. De aceea, nu aveţi dreptul să folosiţi imprimanta decât pentru a imprima lucrările, referatele, etc. realizate de Dvs. în cadrul activităţilor de laborator sau de elaborare a lucrării de diplomă. Este interzisă imprimarea documentelor de pe Internet, a cursurilor, etc. fără aprobarea şefului laboratorului, prof. Daniel Ioan.

Administratori de reţea:

Bogdan Brătucu (`help@lmn.pub.ro`, `bbg@lmn.pub.ro`)

Victor Stănescu (`help@lmn.pub.ro`, `bruno@lmn.pub.ro`)

Anexa C

Evoluţia Laboratorului de Metode Numerice – LMN

Istoria Laboratorului de Metode Numerice - LMN a început în decembrie 1983, când în sala EB–206 a noului local al Politehnicii Bucureştene (pe atunci I.P.B.) s-a instalat primul terminal "la distanţă" la sistemul de calcul FELIX C256 al Centrului de Calcul IPB operat cu pachetul de teleprelucrare "SESAM". Acest teminal era de tip Centronix, un fel de teletype, cu tastatură si imprimare pe hârtie "de calculator". Chiar dacă acum este un centru de cercetare şi învăţământ avansat în domeniul utilizării profesionale a calculatoarelor în ingineria electrică, laboratorul şi-a păstrat numele tradiţional, provenit de la cursul de "Metode Numerice", oferit studenţilor din al II-lea an al Facultăţii de Electrotehnică, începând din 1990.

Dotarea şi preocupările laboratorului au urmărit dezvoltarea impresionantă a tehnologiei informaţiei, marcând câteva evenimente, premiere naţionale. Trebuie menţionate: instalarea primelor calculatoare cu sistem de operare CP/M pentru microprocesoare pe 8 biţi –8080 şi Z80– (FELIX M118, TPD în 1985), primele calculatoare cu procesoare de 16 biţi (8086) ce utilizau sistemul de operare MS-DOS (FELIX PC şi, din 1989, compatibile cu IBM PC-AT), trecerea prin toate formatele de dischete flexibile de la 8" şi 5" până la $3^{1/2}$", dar şi conectarea în 1984, prin linie telefonică dedicată de peste 10 km, a unor teminale inteligente la puternicul –pentru acea vreme– minicalculator WANG VS80, aflat în proprietatea ICPE.

Tehnologiile de imprimare au evoluat şi ele, de la imprimanta cu rozetă Robotron la imprimanta cu ace şi, din 1990, la imprimanta cu laser. În aceste condiţii s-au dezvoltat produse program pentru mai multe aplicaţii, cum sunt: DISTRIB (analiza circuitelor electrice cu parametri distribuiţi, dezvoltat în mare parte pe un sistem de calcul foarte puternic de tip CRAY la Electricité de France – DER, începând încă din anul 1978), TEZED (program de editare a tezaurului de termeni ai Comisiei Electrotehnice Internaţionale, executabil pe WANG în 1985), LOCAP (program de analiză la nivel de porţi a circuitelor logice, combinaţionale şi secvenţiale, dezvoltat sub CP/M în perioada 1985-1988) şi FAP (program pentru analiza câmpurilor 2D, dezvoltat iniţial sub MS-DOS, începând din 1988).

Un eveniment cu implicaţii majore în viaţa laboratorului l-a constituit derularea din 1991 a proiectului TEMPUS JEP 2717, finanţat de Comisia Europeană cu peste 600.000 ECU şi dedicat înfiinţării unei şcoli postuniversitare de inginerie electrică asistată de calculator, cu două direcţii: CAD electromagnetic şi VLSI. În acest proiect au fost implicate cele mai importante personalităţi europene din domeniul calculului de câmp electromagnetic şi a fost evaluat ca proiect model, continuând în perioada 1995-1996 ca reţea de excelenţă europeană – TEMPUS JEN 2717. Folosind bugetul acestui proiect, s-a realizat saltul din lumea DOS în universul UNIX, realizându-se în 1992, pentru prima dată în ţară, o reţea locală de staţii grafice, cu caracter hibrid, bazată pe tehnologia client/server, care încorpora (în sala EB-213) platforme produse de Hewlett Packard (HP 9700), Digital (DEC Station 5000), Sun (SPARC Station) şi Silicon Graphics (Iris Indigo). Legătura la Internet, în 1993, prin proiectul Technische Hochschule Darmstadt – THD la care laboratorul a contribuit activ, a reprezentat o altă deschidere majoră. Accesul la reţeaua RoEduNet şi, prin intermediul ei, la conectivitate Internet completă, constituie una din cele mai importante resurse disponibile în LMN.

Pachetele de programe achiziţionate pentru aceste platforme (sistemele de operare de tip Unix, mediile de dezvoltarea programelor, pachetele de analiză a câmpului electromagnetic FLUX2D, MSC-EMACS şi MEGA, precum şi pachetele de proiectare VLSI: Cadence, HILO, HSPICE şi SOLO) şi documentaţia care le însoţea a avut un impact major atât în activitatea de cercetare, cât şi în cea de învăţământ avansat (pregătirea doctoranzilor). Cu această ocazie s-au proiectat şi realizat practic (prin serviciul EUROCHIP, din cadrul programului ESPRIT al Uniunii Europene, la care ne-am afiliat din 1992) primele circuite integrate VLSI în tehnologia CMOS de 1μm. Lista principalelor pachete software instalate în LMN este prezentată în anexa D.2.

Schimbul de profesori şi studenţi în ambele sensuri a condus la recunoaşterea internaţională a laboratorului, nu numai în marile universităţi europene, ci şi în SUA şi în Japonia, unde avem parteneri în peste 20 laboratoare universitare de mare prestigiu. La aceasta a contribuit, pe lângă proiectele educaţionale şi de cercetare comune, şi organizarea în România a două importante evenimente ştiinţifice: Conferinţa Internaţională de Educaţie Asistată de Calculator a Inginerilor – CAEE'93 (cu 120 participanţi din 20 ţări) şi primul seminar Româno-Japonez în domeniul electromagnetismului şi mecanicii aplicate RJJSAEM'96 (cu 30 participanţi japonezi).

LMN este printre primele entităţi din ţară care şi-au realizat propria pagină de Web (în 1994), în care, în premieră naţională au fost postate în 1996 materiale didactice interactive, dezvoltate în Java.

Un alt proiect care a contribuit esenţial la dezvoltarea LMN îl constituie proiectul TEMPUS S-JEP 9122 cu titlul *Comunicare Audiovizuală bazată pe Tehnologia Informaţiei*, finanţat de Comisia Europeană şi de Fundaţia Soros pentru o Societate Deschisă. În cadrul acestui proiect, desfăşurat în perioada 1995-1998, s-a dezvoltat laboratorul de comunicare profesională (EB- 212) bazat pe prima reţea Fast Ethernet din U.P.B., care încorporează calculatoare Multimedia PC-compatibile (cu procesoare Intel Pentium şi 32 MB RAM) şi două servere, unul Windows NT 4.0 şi altul Linux. Tot în cadrul acestui proiect s-a obţinut din partea Hogeschool van Utrecht o donaţie de 17 staţii grafice (DEC

VAX), folosite ca Xterminale în laboratorul de inginerie matematică (sala EB-210) precum şi 12 calculatoare PC ce completează reţeaua de 12 calculatoare IBM PS/2 (EB-207) finanţată de Ministerul Educaţiei Naţionale şi folosită în mod curent de studenţii din anul II, la disciplina de Metode Numerice. Lista completă a echipamentelor instalate în LMN este prezentată în anexa D.1.

O importantă resursă a LMN pusă la dispoziţia studenţilor şi cadrelor didactice este Centrul de documentare şi training (EB-206) în care sunt disponibile peste 500 cărţi şi ultimele numere din 32 de publicaţii ştiinţifice din domeniile: matematici aplicate, calculatoare, câmp electromagnetic şi circuite electronice. Lista bibliografică a acestui manual reprezintă doar o parte din documentaţia pe care o puteţi consulta (din păcate fără a o putea împrumuta) în cadrul laboratorului.

Iniţiată încă din 1995 de LMN, *Iniţiativa Naţională pentru Calcule de Înaltă Performanţă* urmăreşte instalarea în reţeaua *RoEduNet* a primului supercalculator din ţară (care sperăm să fie menţionat cât mai curând în celebra listă TOP500 redactată semestrial de Jack Dongarra şi care conţine cele mai puternice calculatoare din lume). Realizarea, împreună cu alţi parteneri, a proiectului "CoLaborator – Bază de Cercetare cu Utilizatori Multipli în Domeniul Calculelor de Înaltă Performanţă", finanţat începând cu anul 1997 de către Banca Mondială, dă speranţa că acest deziderat va fi îndeplinit.

În afara acestui proiect, LMN este implicat în mai multe proiecte europene sau internaţionale de cercetare. Acest lucru ar fi imposibil fără permanenta împletire cu activitatea educaţională.

LMN organizează anual cicluri de "autoinstruire" în utilizarea calculatoarelor, deschise studenţilor din anii mici ai tuturor facultăţilor din UPB. Cei mai talentaţi studenţi sunt ulterior implicaţi în proiectele de cercetare ale laboratorului.

În afara cursului de Metode Numerice, LMN oferă mai multe cursuri în anii mari ai secţiei de Inginerie Electrică Asistată de Calculator de la Facultatea de Electrotehnică şi organizează anual pregătirea aprofundată (anul VI) pentru 20 studenţi ce provin de la diferite facultăţi de profil electric din ţară (electrotehnică, electronică, automatică, energetică sau calculatoare), direcţiile de specializare fiind "CAD electromagnetic" şi microelectronică. În plus, oferă un curs în limba engleză pentru Departamentul de Ştiinţe Inginereşti din UPB.

În laborator îşi desfăşoară cercetările cca. 10 doctoranzi în ingineria electrică (în domeniile: analiza numerică a câmpului electromagnetic, optimizarea dispozitivelor electromagnetice, algoritmi paraleli şi distribuiţi, calcule de înaltă performanţă, efecte de câmp în circuite electronice, defectoscopie electromagnetică nedistructivă, ştiinţă şi inginerie computaţionale, educaţie asistată de calculator, tehnici numerice în ingineria electrică, proiectarea asistată de calculator în ingineria electrică, proiectare electronică automată, analiza circuitelor cu parametri distribuiţi, elemente electromagnetice de circuit, analiza sistemelor cu microunde, probleme fizice cuplate, etc.) şi sunt primiţi în stagiu studenţi şi doctoranzi din ţară (Iaşi, Cluj, Braşov, etc.) şi din străinătate (Franţa, Marea Britanie, Germania, Spania, etc.).

Cercetarea ştiinţifică desfăşurată în laborator este finanţată de Ministerul Educaţiei Naţionale (CNCSU şi CNFIS prin proiectele Băncii Mondiale), Ministerul Cercetării

şi Tehnologiei, industria din România, instituţii şi organizaţii din stăinătate (proiecte PECO, COPERNICUS sau proiecte bilaterale, ca RJSAEM).

În ciuda aparenţei, LMN este un "laborator deschis" (a nu se confunda cu necesitatea respectării unor reguli stricte de acces, securitate şi comportament atunci când se operează într-un mediu tehnologic avansat şi complex!), oferind suport şi acces la resurse oricărui membru din comunitatea academică (şi cu precădere celor talentaţi, ambiţioşi şi creativi) cu singurele condiţii ca acesta să aibă un proiect clar (obiective, termene, metode de abordare şi un buget), să respecte regulamentul laboratorului şi să-şi asume obligaţia morală de a menţiona sprijinul LMN – "acknowledgment" în documentele elaborate în acest proiect. Mediul realizat în LMN (echipamente, software, documentaţie, servicii, competenţe interdisciplinare) este similar cu cel al celor mai importante laboratoare similare din lume şi crează condiţiile emulaţiei şi creativităţii specifice mediului academic. Fără caracterul său interdisciplinar, deschis intensei cooperări internaţionale, acest mediu n-ar putea fi menţinut. Dacă accepţi exigenţele internaţionale asupra nivelului necesar cercetării ştiinţifice universitare, poţi contribui şi tu la succesul laboratorului şi implicit la succesul tău în carieră.

Credem că prezintă interes câteva din concluziile experienţei LMN în utilizarea tehnologiei avansate a informaţiei:

- O importanţă deosebită în cercetare o au instrumentele folosite, acestea trebuie să fie dintre cele mai puternice şi eficiente, efortul făcut pentru învăţarea utilizării lor profesionale fiind din plin răsplătit. Mediul folosit trebuie să fie totuşi relativ stabil, deoarece schimbarea prea rapidă a versiunilor sau tehnicilor scade eficienţa cercetării.

- Achiziţionarea de licenţe software este la fel de importantă ca achiziţionarea de echipamente (hardware); rezultatele au fost mult mai bune rezervând o mare parte a bugetului pentru software, datorită accesului la documentaţie şi la ultimele versiuni ale pachetelor de programe, lucru ce a modificat mentalitatea utilizatorilor şi a crescut productivitatea lor.

- Un impact deosebit în utilizarea tehnicii moderne de calcul îl are conectarea la reţeaua Internet; în acest fel sunt disponibile informaţii extrem de valoroase care altfel ar fi fost foarte dificil sau imposibil de obţinut, dar folosirea Internet-ului nu trebuie făcută numai din poziţia de consumator, ci prin producerea de informaţie generată local se uşurează parteneriatul şi integrarea cercetării româneşti în ansamblul european şi global.

- Ingineria electrică are în prezent un puternic caracter interdisciplinar (electrotehnică, electronică, sisteme automate, calculatoare) şi se bazează pe solide cunoştinţe de matematică. Deciziile strategice privind temele de cercetare şi tehnicile ce vor fi folosite trebuie să fie luate astfel încât să se "intuiască" valul domeniului încă înainte ca acesta să ajungă la vârf, când e prea târziu de recuperat decalajul. Păstrarea echilibrului între tradiţie şi migrarea către temele moderne finanţate generos reprezintă secretul unui bun management ştiinţific. Deoarece numărul ideilor

esenţialmente noi este mai mic decât s-ar părea la prima vedere, valorificarea creatoare a unei tradiţii solide generează suficientă flexibilitate şi asigură succesul scontat.

- Componenta umană este mult mai importantă decât orice calculator, software sau documentaţie; ea este încurajată prin lucrul în echipe interdisciplinare, sprijinirea ideilor tinerilor talentaţi, crearea unei emulaţii specifice mediilor academice, respectarea libertăţii academice, a creativităţii şi iniţiativei individuale, contactele personale cu partenerii din străinătate, identificarea şi participarea la viaţa unei comunităţi ştiinţifice de calitate prin participarea la evenimentele periodice (din străinătate, dar şi atragerea lor în România!), proiecte comune şi valorificarea rezultatelor în publicaţiile ştiinţifice.

- România are o capacitate limitată de a absorbi resursele financiare disponibile pe plan mondial pentru cercetare şi acest lucru se datoreşte deficitului de deprinderi manageriale moderne şi mentalităţilor deplorabile (credem că respectarea principiului "minimei seriozităţi" este la fel de importantă sau poate chiar mai importantă decât "*minima moralia*" sau decât aspectele juridice, economice sau politice).

Ne exprimăm încrederea că istoria LMN nu se va opri aici. Ultimele noutăţi le puteţi afla din pagina noastră de Web: `http://www.lmn.pub.ro`.

Anexa D

Resursele Laboratorului de Metode Numerice

D. 1 Resurse hardware

D. 2 Resurse software

LMN îşi desfăşoară activitatea în sălile:

- EB-206 - Centrul de documentare şi training
- EB-207 - Laboratorul de iniţiere în metode numerice
- EB-210 - Laboratorul de inginerie matematică
- EB-211 - Secretariat, relaţii cu publicul
- EB-212 - Laboratorul de comunicare profesională
- EB-213 - Laboratorul de inginerie electrică asistată de calculator

D.1 Resurse hardware

Reţeaua de calculatoare a LMN (subdomeniul `lmn.pub.ro`) are topologia din figurile D.1 – D.4.

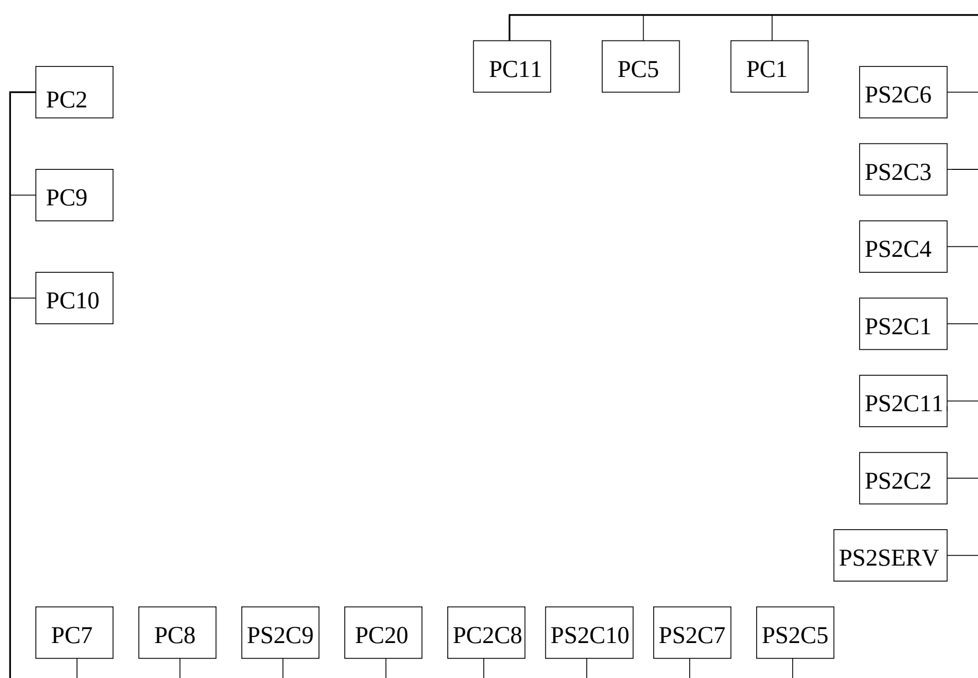


Fig. D.1: Topologia reţelei EB-207

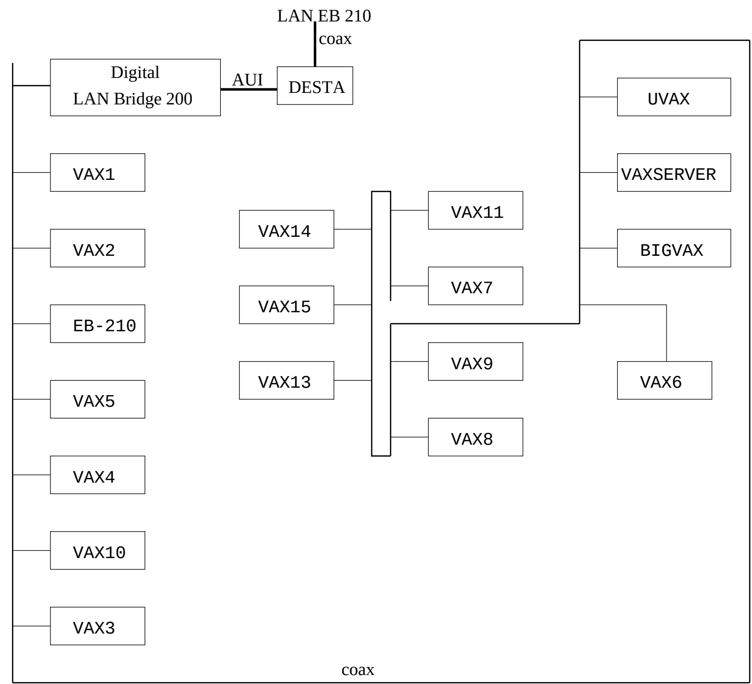


Fig. D.2: Topologia reţelei EB-210

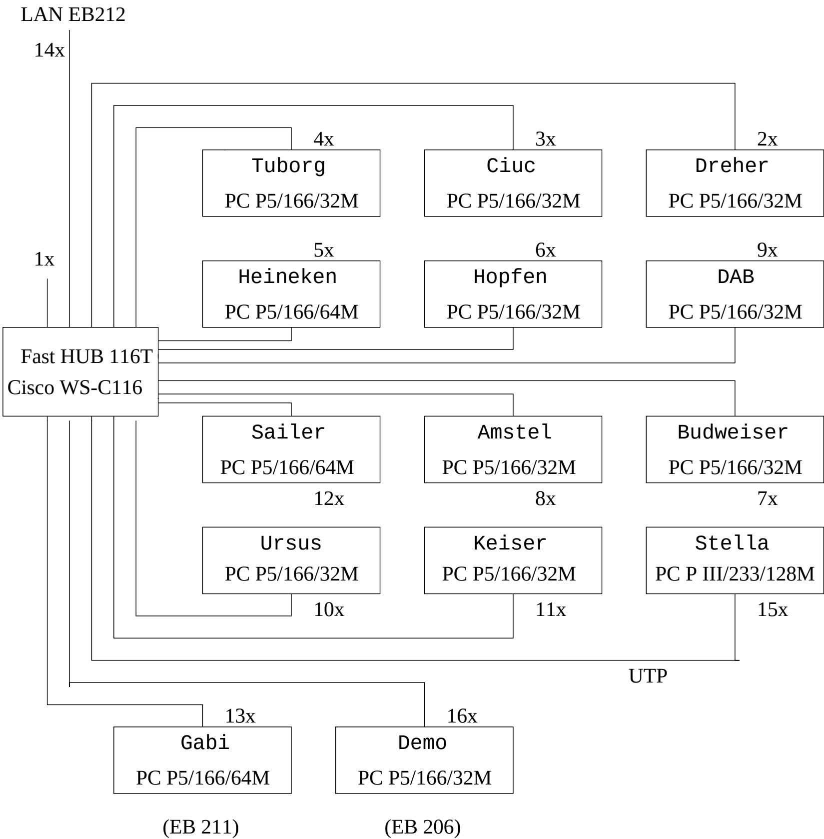


Fig. D.3: Topologia reţelei EB-212

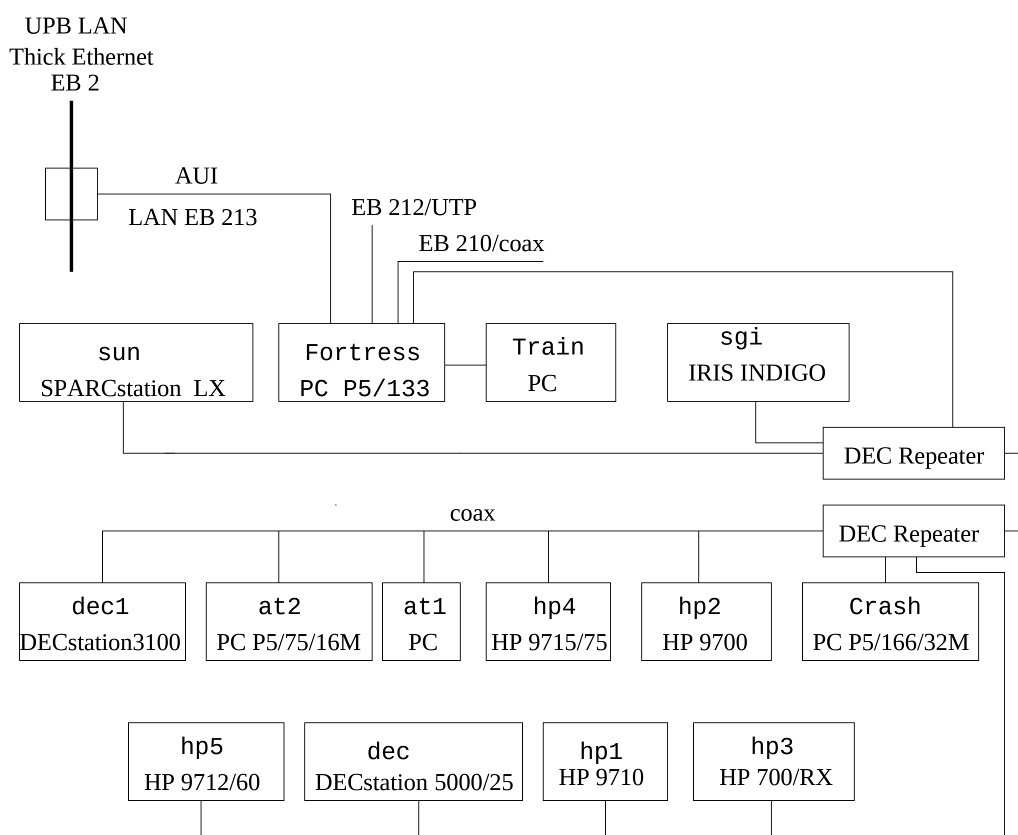


Fig. D.4: Topologia reţelei EB-213

Principalele caracteristici ale sistemelor instalate sunt prezentate în tabelul de pe paginile următoare.

Nume sistem	**Model**	**Sala**	**Procesor**	**Monitor**	**RAM** [MB]	**HDD** [B]	**Periferice**	**Adresa IP** 141.85.34.*xxx*
lmndec	DEC Station 5025	EB-213	RISC	color 21”	40M	820M	95Mb TK tape	4
lmnhp1	HP 9710	EB-213	RISC	color 21”	32M	9 GB	CD-ROM	2
lmnhp2	HP 9705	EB-213	RISC	gray 19”	24M	140M	2 GB DAT tape	1
lmnhp3	HP 700/RX	EB-213	RISC	gray 19”	–	–	–	3
lmnhp4	HP 9715/75	EB-213	RISC	color 21”	64M	540M	HP LaserJet 4	7
lmnhp5	HP 9712/60	EB-213	RISC	color 19”	32M	1G	–	10
lmnsgi	SGI-IRIS INDIGO	EB-213	RISC R4000	color 21”	32M	2.08G	CD-ROM	5
lmnsun	Sun SPARC Station LX	EB-213	RISC SPARC	color 21”	32M	270M	CD- ROM, 150 Mb QIC tape	6
Crash	PC	EB-213	Pentium/166	color 17”	32M	5.8G	CD-ROM	56
at1	PC	EB-213	Pentium/75	color 17”	16M	400M	–	11
at2	PC	EB-213	Pentium/75	color 17”	16M	400M	–	12
Train	PC	EB-213	Pentium/75	color 17”	16M	1.2G	–	8
Fortress	PC	EB-213	80486/100	monocrom 14”	32M	420M	–	129, 9, 193, 65, 34
Sailer	PC	EB-212	Pentium/166	color 17”	64M	4G	Epson Stylus color 3000, MPC	142
Amstel	PC	EB-212	Pentium/166	color 17”	32M	1624M	MPC	138
Dreher	PC	EB-212	Pentium/166	color 17”	32M	1624M	MPC	132
Kaiser	PC	EB-212	Pentium/166	color 17”	32M	1624M	MPC	141
Ursus	PC	EB-212	Pentium/166	color 17”	32M	1624M	MPC	140
Ciuc	PC	EB-212	Pentium/166	color 17”	32M	1624M	MPC	133
DAB	PC	EB-212	Pentium/166	color 17”	32M	1624M	MPC	139
Budweiser	PC	EB-212	Pentium/166	color 17”	32M	1624M	MPC	137
Hopfen	PC	EB-212	Pentium/166	color 17”	32M	1624 M	MPC	135
Tuborg	PC	EB-212	Pentium/166	color 17”	32M	2.1G	MPC	134

Nume sistem	**Model**	**Sala**	**Procesor**	**Monitor**	**RAM** [MB]	**HDD** [B]	**Periferice**	**Adresa IP** 141.85.34.*xxx*
Stella	PC	EB-212	Pentium II/233	color 21”	128M	9G	Scanner Umax Astra 1200, CD-ROM REC, Video	145
Heineken	PC	EB-212	Pentium/166	color 21”	64M	7G	HP LaserJet 5MP, Jaz	131
Gabi	PC	EB-211	Pentium/166	color 17”	64M	1.3 G	HP LaserJetIIP	143
Demo	PC	EB-206	Pentium/166	color 17”	32M	1624 M	MPC	146
EB-210	PC	EB-210	Pentium/166	monocrom 14”	64M	4 G	–	199
VAX1	VAX Station 3100	EB-210		color 21”	16M	120M	–	201
VAX2	VAX Station 3100	EB-210		color 21”	16M	120M	–	202
VAX3	VAX Station 3100	EB-210		color 21”	16M	120M	–	203
VAX4	VAX Station 3100	EB-210		color 21”	16M	120M	–	204
VAX5	VAX Station 3100	EB-210		color 21”	16M	120M	–	205
VAX6	VAX Station 3100	EB-210		color 21”	16M	120M	–	206
VAX7	VAX Station 3100	EB-210		color 21”	16M	120M	–	207
VAX8	VAX Station 3100	EB-210		color 21”	16M	120M	–	208
VAX9	VAX Station 3100	EB-210		color 21”	16M	120M	–	209
VAX10	VAX Station 3100	EB-210		color 21”	16M	120M	–	210

Nume sistem	**Model**	**Sala**	**Procesor**	**Monitor**	**RAM** [MB]	**HDD** [B]	**Periferice**	**Adresa IP** 141.85.34.*xxx*
VAX11	VAX Station 3100	EB-210		color 21”	16M	120M	–	211
VAX13	VAX Station 4000/60	EB-210		color 21”	16M	120M	–	213
VAX14	VAX Station 4000/60	EB-210		color 21”	16M	120M	–	214
VAX15	VAX Station 4000/60	EB-210		color 21”	16M	120M	–	215
VAXSERVER	Microvax 3800	EB-210		monocrom 14”	16M	240M	Scanner	212
UVAX	Microvax II	EB-210		monocrom 14”	–	–	DEC Printer	222
BIGVAX	Microvax 4800	EB-210		monocrom 14”	–	–	CD-ROM	200
PS2 server	IBM PS/2 65SX	EB-207	286	monocrom 12”	2/4M	105 Mb	Printer IBM PS/1	-
PS2C1	IBM PS/2/30	EB-207	286	monocrom 12”	2M	–	–	–
PS2C2	IBM PS/2/30	EB-207	286	monocrom 12”	2M	–	–	–
PS2C3	IBM PS/2/30	EB-207	286	monocrom 12”	2M	–	–	–
PS2C4	IBM PS/2/30	EB-207	286	monocrom 12”	2M	–	–	–
PS2C5	IBM PS/2/30	EB-207	286	monocrom 12”	2M	–	–	–
PS2C6	IBM PS/2/30	EB-207	286	monocrom 12”	2M	–	–	–
PS2C7	IBM PS/2/30	EB-207	286	monocrom 12”	2M	–	–	–
PS2C8	IBM PS/2/30	EB-207	286	monocrom 12”	2M	–	–	–
PS2C9	IBM PS/2	EB-207	286	monocrom 12”	2M	–	–	–
PS2C10	IBM PS/2/30	EB-207	286	monocrom 12”	2M	–	–	–
PS2C11	IBM PS/2/30	EB-207	286	monocrom 12”	2M	–	–	–
PC1	Dec Station 300 digital	EB-207	386SX	color 14”	4M	40 Mb	–	–
PC2	Digital	EB-207	386	color 14”	4M	–	–	–
PC5	Dec Station 320 digital	EB-207	386	color 14”	2M	115 Mb	–	–

Nume sistem	Model	Sala	Procesor	Monitor	RAM [MB]	HDD [B]	Periferice	Adresa IP 141.85.34.xxx
PC7	Digital 320	EB-207	386	color 14”	6M	115 Mb	–	–
PC8	PC 655B3	EB-207	386	color 14”	3M	40 Mb	–	–
PC9	Digital 316	EB-207	386	color 14”	3M	40 Mb	–	–
PC10	Digital 316	EB-207	386	color 14”	5M	–	–	–
PC11	Dec Station 320 digital	EB-207	386	color 14”	4M	–	–	–
PC20	Tulip pc	EB-207	386SX	monocrom 12”	2M	80 Mb	–	–

D.2 Resurse software

Pe sistemele prezentate în figurile D.1 — D.4 sunt instalate mai multe pachete de programe, multe dintre ele accesibile în reţea, dintre care cele mai importante sunt:

1. **Software de sistem**
   - sistem de operare HP-UX pentru staţiile grafice `hp`
   - sistem de operare Ultrix pentru staţiile grafice `dec`
   - sistem de operare Sun OS pentru `sun`
   - sistem de operare Irix pentru `sgi`
   - sistem de operare Linux pentru PC din EB-206, EB-210, EB-212 şi EB-213 (versiunea 2.0.35 la data publicării)
   - interfaţa grafică (GUI) de tip XWindow/Motif pentru toate sistemele de tip Unix
   - sistem de fişiere distribuit NFS pentru toate maşinile
   - conexiune Unix-PC prin Digital Pathworks şi PC-NFS
   - sisteme de operare MS Windows NT (server şi workstations) pentru PC (EB-212)
   - sisteme de operare MS-DOS, MS Windows 3.1 pentru PS/2 (EB-207)
   - NetWare 3.11, sistem pentru reţeaua PS/2 (EB-207)
2. **Software de dezvoltare**
   - Compilator Ultrix C (`dec`)
   - Compilator Sun C (`sun`)
   - Compilator HP-UX C/C++ (`hp1`)
   - Compilator Dec Fortran (`dec`)
   - Compilator HP-UX Fortran77 (`hp1`)
   - Compilatoarele GNU C, C++, Fortran v2.8.1 (pentru toate sistemele Unix)
   - Compilator Turbo Pascal v6.0/Windows
   - Compilator Turbo C++ v3.0 /Windows
   - Editoare `vi` şi `emacs` v2.0.4 pe toate sistemele Unix
   - Alte editoare de text specifice fiecărui sistem ca: `Vue-Pad (hp)`, `XPSView (sgi)`
   - Medii de depanare şi dezvoltare: `gdb` (Linux, `hp4`, `hp5`), `xxgdb`, `ddd` v2.2 (Linux), `dbx` (`dec`, `sgi`, `sun`), `ctrace` (`dec`, `sgi`), `xdb` (`hp1`), `gldebug` (`sgi`), `xwpe` (Linux), `RHide` (Linux, în mod text)
   - HP-UX C/C++ Developer's kit (`hp1`)
   - Interpretoare scripturi: Tc/Tkl, Java

- Biblioteci matematice sursă Fortran (NAG, Linpack, Eispack, acces la Netlib)
- Biblioteci grafice (Xlib, Sun-Direct Xlib, Open GL, Starbase-HP, PIX-HP)
- Medii CASE generatoare ferestre: DEC Vuit, DEC Fuse (`dec`)
- Biblioteci de funcţii pentru calcule distribuite: PVM, MPI, PETSc CORBA, LAM, MPICH, NetSolve (EB-212)

3. **Software matematic şi ingineresc**
   - Mathematica (2 licenţe pe `sgi`)
   - Scilab (EB-210, 212, 213)
   - MATLAB, SIMULINK, LabVIEW V3.1 Student Ed., MAPLE sub MS-Windows
   - CANM - pachet de programe pentru instruirea asistată de calculator în domeniul metodelor numerice dezvoltat în LMN (EB-207)

4. **Analiza şi proiectarea dispozitivelor electromagnetice (CAD, FEM)**
   - AutoCAD pentru `hp`, `sgi`, `sun`, MS-Windows
   - MSC/EMAS - MacNeil Schwendler - Analiza numerică a câmpului electromagnetic 3D (`hp1`)
   - FLUX2D - CEDRAT - Analiza numerică a câmpului electromagnetic 2D (`hp1`)
   - MEGA - Donat de Bath University - Analiza câmpului electromagnetic 3D (`hp`)
   - FAP - dezvoltat în LMN pentru Analiza câmpurilor 2D, disponibil sub Windows şi Linux

5. **Proiectarea şi analiza microsistemelor (VLSI)**
   - CADENCE - mediu integrat pentru proiectarea circuitelor VLSI care permite descrierea comportamentală (Hardware Description Language) şi structurală, sinteza (inclusiv de nivel înalt), extragerea parametrilor, simularea sistemului şi editarea măştilor (o lincenţă Edge pe `dec` şi o licenţă OPUS pe `hp`)
   - Licenţe pentru tehnologiile European Silincon Structure (ES2): 1.5, 1.0, 0.7 μm
   - SOLO 1400 - mediu integrat pentru proiectarea rapidă a circuitelor ASIC în tehnologiile ES2 (10 licenţe pe `sun`)
   - HILO produs de GENRAD, simulator pentru proiectarea logică (`hp` şi `dec`)
   - HSPICE produs de META - simulator de circuite la nivel de tranzistori (licenţe pe `hp`, `dec`, `sun`)
   - CATHEDRAL, mediu pentru proiectarea sistemelor SDP (Digital Signal Processing) (`hp2`)
   - ALTERA MAX2PLUS - mediu integrat pentru proiectarea ariilor de porţi logice programabile - FPGA (HP şi MS-Windows)

- XILINX - mediu integrat pentru proiectarea ariilor de porţi programabile FPGA (2 licenţe pe MS-Windows)
- Medii de proiectare sub Linux: CACD, CHIPMUNK, MAGIC, OCEAN, UCB–TOOLS, ALLIANCE, SIS, LLL, VERIWELL (EB212)
- SPICE - Berkeley Free Software (3e2) pe `dec` şi Linux
- Pspice - demo sub MS Windows pentru reţeaua PS/2 (EB-207)
- ELCUT - analizor circuite electrice (EB-207)

6. **Software de comunicare**
   - Poşta electronică (elm, pine, MH, mailx)
   - Browsere Web şi Internet (Mosaic , Netscape)
   - Alte instrumente pentru explorare Internet (Xgopher, Xarchie)
7. **Software pentru vizualizarea ştiinţifică, pregătirea documentelor şi producţie multimedia**
   - Pregătire grafice (GNUplot, IRIS Explorer)
   - Pregătire figuri (Xfig, XPaint, GIMP 1.0)
   - Pregătire documente (TeX, LaTeX, BibteX, GhostView, LaTeX2html)
   - Pregătire prezentări şi documente electronice (IRIS Showcase, Acrobat Reader, x11amp, raplayer sub Linux, iar sub MS-Windows: MS–Office97 Professional Ed. Romanian Vers., Word97, Excel97, Powerpoint97, Access97, MS Works, Corell Draw 6.0, 3D Studio MAX, Adobe Photoshop 3.0, Adobe PhotoDeluxe, Adobe Premiere 4.2, Cubasis Audio 1.6, Macromedia Director 4.0, Acrobat Reader 3.0, Visual Cafe, Easy CD-PRO, Miro VIDEO, Miro 3DFX,

O mare parte a comenzilor şi instrumentelor din lista de mai sus sunt descrise pe parcursul lucrării.

D.3 Interfaţa grafică a butoanelor mouse-ului

Apelul pachetelor software descrise în anexa D.2 poate fi realizat utilizând interfaţa grafică (click cu butoanele mouse-ului pe fundalul ecranului).

În cele ce urmează sunt prezentate meniurile de comenzi disponibile prin intermediul mouse-ului sub Linux şi pe staţiile grafice.

D.3.1 Meniuri de comenzi disponibile sub Linux

Meniul "Toolbox" (fig. 1.2) permite accesul la:

- Terminal nou

- NetScape
- Utilitare pentru gestionarea fişierelor
  - TkDesk
  - Midnight Commander (emulează Norton Commander)
  - Xfilemanager
  - Xfm
- Blocarea ecranului
- Calculator de buzunar
- Captură grafică de ferestre
- Vizualizarea încărcării sistemului
- Distrugere fereastră
- Spaţii de lucru
- Poştă electronică

Meniurile de comenzi (fig. D.5) disponibile sub Linux şi accesibile prin click cu butonul stâng pe fundal sunt (EB-210, EB-212):

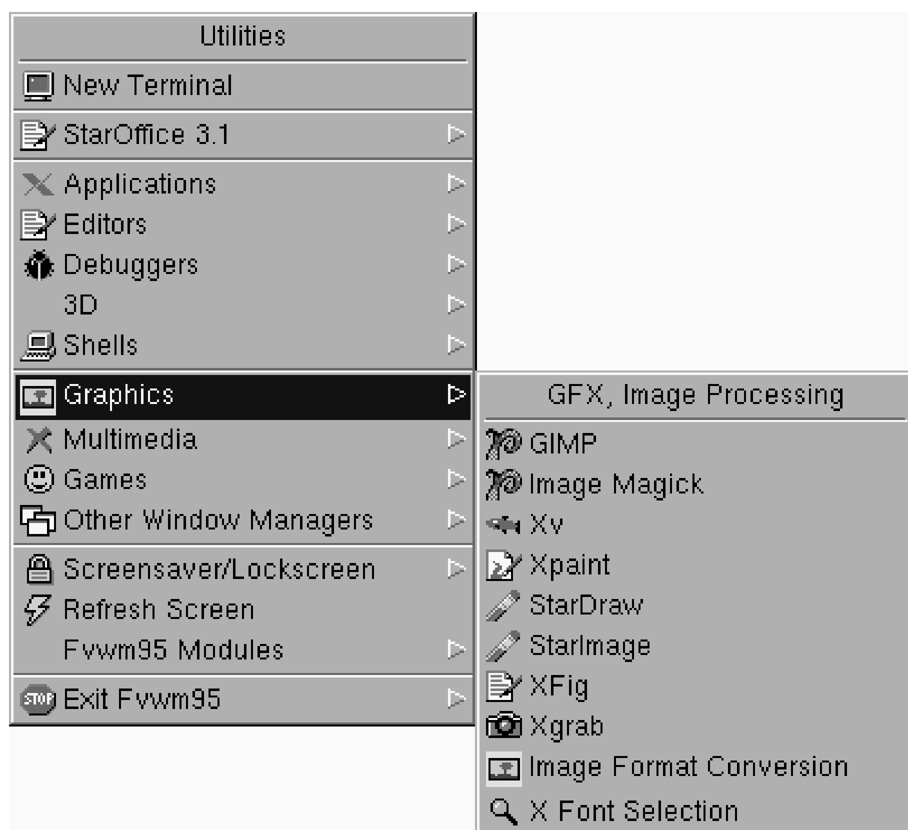


Fig. D.5: Meniul principal Linux

- Terminal nou
- Star Office

  - Editor texte - Star Writer
  - Editor desene - Star Draw
  - Editor figuri - Star image
  - Calculator - Star Calc
  - Grafice - Star Chart
  - Editor ecuaţii - Star Math
- Aplicaţii
  - Browser Web
    * Netscape 3.04
    * Netscape 4.03
    * Lynx 2.8 (text)
  - Editoare
  - Depanatoare
  - Cititor PDF
  - Cititor PS
  - Editor bitmap
  - Documentare Man
  - Calendar
- Editoare
  - XEmacs
  - vi
  - xvi
  - joe
  - xedit
- Depanatoare
  - DDD
  - Manual DDD
  - Xxgdb
  - GNU gdb
- 3D
  - Editor Scene VRML (sced)
  - Manual sced
  - Tutorial sced
  - Modelare 3D (nnm)

- Shelluri
  - XTerm
  - Rxwt (VT100)
  - ETerm
- Grafică
  - GIMP
  - Image Magick
  - XV
  - Xpaint
  - XFig
  - Xgrab
  - Conversie format imagine
  - Selecţie fonturi X
- Multimedia
  - player
  - CD
  - audio mixer
  - MPEG 3
  - efecte vizuale
- Jocuri
- Managere de ferestre
- Blocare ecran, fundal ecran
- Reiniţializare ecran
- Module Fvwm95
- Ieşire din sesiune

D.3.2 Meniuri de comenzi disponibile în reţeaua de staţii grafice

Meniurile de comenzi disponibile în reţeaua heterogenă de staţii grafice cu procesoare RISC permit accesul de la distanţă, în tehnologia client/server. Meniuri distincte sunt mapate pe cele trei butoane ale mouse-ului.

Meniuri de comenzi mapate pe butonul stâng

Prin click cu butonul stâng pe fundalul ecranului sunt disponibile următoarele comenzi (Instrumente generale – fig. D.6):

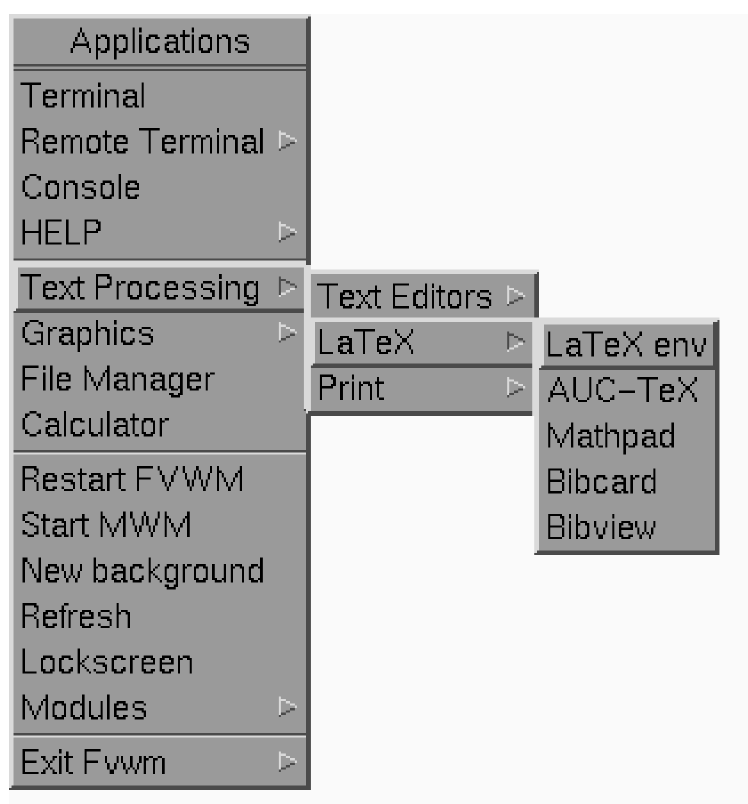


Fig. D.6: Exemplu de meniu asociat butonului stâng

- Terminal nou
- Terminal pe altă maşină
  - lmnhp1
  - lmnhp4
  - lmnhp5
  - lmndec
  - lmnsgi
- Consolă
- Documentaţie
  - HELP
  - MAN
  - LaserROM
  - Insight
  - XMan
  - OpenBook
- Procesoare texte şi documente

  - Editoare texte
    * Emacs
    * VeEdit
    * TextEdit
    * Asedit
  - LaTeX
    * Mediul LaTeX
    * AUC-TEX
    * Mathpad
    * Bibcard
    * Bibview
  - Imprimare
    * fişier la imprimantă
    * fereastră la imprimantă
    * fereastră în fişier
- Grafică
  - XFig
  - Ghostview
  - Adobe PS
  - Acrobat PDF
  - XV
  - XPaint
  - Xmonitory
- Calculator
- Restart sistem ferestre (FVWM)
- Start manager ferestre MWM
- Fundal ecran nou
- Reiniţializare ecran
- Module FVWM
- Ieşire

Meniuri de comenzi mapate pe butonul central

Butonul din mijloc permite accesul la următoarele comenzi (Instrumente de proiectare – fig. D.7):

- VLSI
  - Cadence OPUS

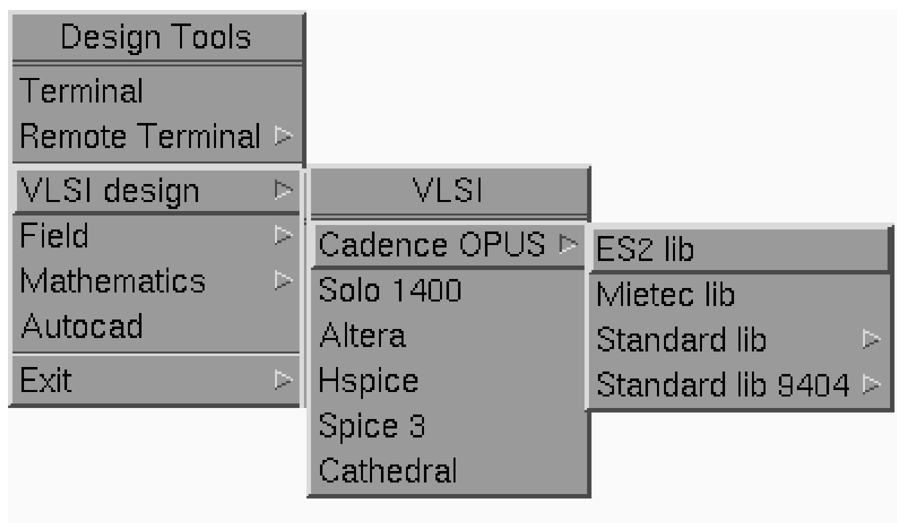


Fig. D.7: Exemplu de meniu asociat butonului central

  - Solo 1400
  - Altera
  - HSpice
  - Spice 3
  - Cathedral
- Câmp
  - Flux2D
  - Mega
  - Emas
- Matematică
  - Mathematica
  - Scilab
  - GNU Plot
- CAD - AutoCAD

Meniuri de comenzi mapate pe butonul din dreapta

Butonul din dreapta accesează comenzile (Instrumente Internet – fig. D.8):

- e-mail
  - elm
  - HPelm

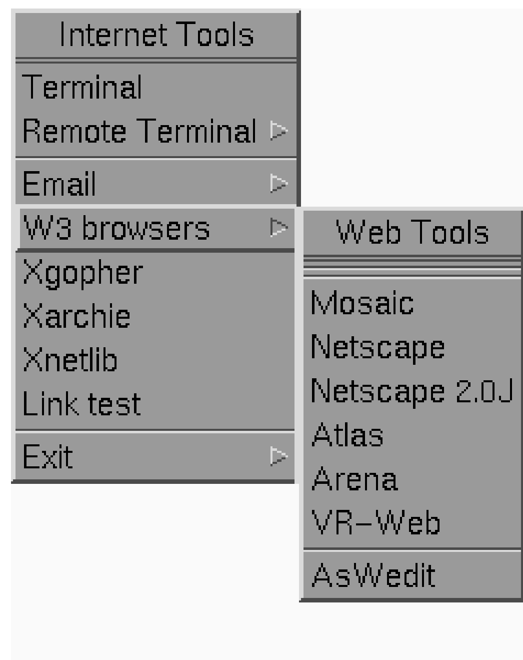


Fig. D.8: Exemplu de meniu asociat butonului din dreapta

  - mailx
  - mailtool
- Browsere W3
  - Mosaic
  - Netscape
  - Netscape 2.0 (Java)
  - Atlas
  - Arena
  - VR-Web
  - AsWedit
- Xgopher
- Xarchie
- Xnetlib
- Test conexiune pub.net

Bibliografie şi Webografie

[1] ***. Elm Info. http://www.myxa.com/elm.html.

[2] ***. Free XWindow system. http://www.xfree86.org/.

[3] ***. Ghostview. http://www.cs.wisc.edu/ ghost/ghostview/index.html.

[4] ***. GIMP – GNU Image Manipulation Program. http://www.gimp.org/.

[5] ***. GNU – Ghostscript. http://www.gnu.org/software/ghostscript/ghostscript.html.

[6] ***. GNU is Not Unix. http://www.gnu.org/.

[7] ***. The GNU plotting utilities. http://www.gnu.org/software/plotutils/plotutils.html.

[8] ***. Internet Architecture Board. http:///www.iab.org/iab/.

[9] ***. ISO IEC POSIX. http://osiris.dkung.dk/JTCI/SC22/WG15.

[10] ***. Linux. http://www.linux.org/.

[11] ***. Linux applications and utilities. http://www.xnet.com/ blatura/linapps.html/.

[12] ***. The Linux encyclopedia. http://www-ocean.tamu.edu/ baum/linuxlist.html/.

[13] ***. Midnight modeller. http://SAL.KachinaTech.COM/E/3/MUM.html.

[14] ***. The Netlib repository. http://www.netlib.org/.

[15] ***. NetScape home page. http://home.netscape.com/.

[16] ***. The Open Group X/Open. http://www.opengroup.org/.

[17] ***. PETSc - The **P**ortable, **E**xtensible **T**oolkit for **Sc**ientific computation. http://www.mcs.anl.gov/pets.

[18] ***. Pine information center. http://www.washington.edu/pine/.

[19] ***. Quadpack. http://www.netlib.org/quadpack/.

[20] ***. Sced – scene editor. http://http.cs.berkeley.edu/ schenney/sced.html.

[21] ***. Scientific applications on Unix. http://sal.kachinatech.com/.

[22] ***. Scilab: A free CACSD package by INRIA. http:///www-rocq.inria.fr/scilab/.

[23] ***. Stardivision. http://stardivision.com/.

[24] ***. TeX Users Group. http://www.tug.org/.

[25] ***. Transactions author information and LaTeX style files. http://www.ieee.org/pubs/authors.html.

[26] ***. The UK TeX archive/ The LaTeX3 project. http://www.tex.ac.uk/.

[27] ***. The VIM (Vi IMproved) home page. http://www.vim.org/.

[28] ***. World Wide Web consortium. http://www.w3.org/.

[29] ***. XFig. ftp://epb1.lbl.gov/xfig.

[30] ***. XPaint. http://www.danbbs.dk/ torsten/xpaint/index.html.

[31] ***. XV. http://www.trilon.com/xv.

[32] ***. *IRIX User's Reference Manual*. Silicon Graphics Inc., 1990.

[33] ***. *The Ultimate Guide to the VI and EX Editors*. Hewlett–Packard Company, 1990.

[34] ***. *Ultrix Reference Pages*. Digital Equipment Co, 1990.

[35] ***. *A Beginner's Guide to HP-UX*. HP 9000 Computers, Hewlett–Packard Company, 1991.

[36] ***. *Mail Systems: User's Guide*. HP 9000 Computers, Hewlett–Packard Company, 1991.

[37] ***. *Text Processing: User's Guide*. HP 9000 Computers, Hewlett–Packard Company, 1991.

[38] ***. *Sun OS 5.1 Reference Manual*. Sun Soft, 1992.

[39] ***. *AutoCAD Release 13 User's Guide*. AutoDesk, 1994.

[40] ***. *Drx. Linux. The Linux Documentation Project*. LSL, 1994.

[41] ***. *Internet Passport*. Prentice Hall, 1995.

[42] J. D. Foley et al. *Computer Graphics. Principles and Practice*. Addison-Wesley, 1987.

[43] V. Cristea et al. *Mai multe despre Internet*. Teora, 1995.

[44] David Baker. Preparing presentation materials using `seminar.sty`. `http://www.comlab.ox.ac.uk/internal/guides/html/present-27/present.html`, 1995.

[45] T. F. Coleman. *Handbook for Matrix Computations*. SIAM, 1988.

[46] Donald A. Curry. *Using C on the UNIX System*. O'Reilly & Associates, Sebastopol, CA, 1989.

[47] E. Cutler. *SCO Unix in a Nutshell*. Romsym Data, 1995.

[48] G. F. Forsythe şi G. B. Moler. *Computer Solutions of Linear Algebraic Systems*. Prentice Hall, 1967.

[49] Open Software Foundation. *OSF/Motif Style Guide*. Prentice Hall, 1991.

[50] D. Frey şi R. Addams. *A Directory of Electronic Mail*. O'Reilly & Assoc., 1994.

[51] Marin Ghinea şi Virgiliu Fireţeanu. *MATLAB Calcul numeric – Grafică – Aplicaţii*. Teora, Bucureşti, 1995.

[52] Daniel Gilly. *Unix in a Nutshell*. O'Reilly, 1992.

[53] Michel Goossens, Frank Mittelbach şi Alexander Samarin. *The LaTeX Companion*. Addison-Wesley, 1994.

[54] John Gustafson. Computational verifiability and feasibility of the ASCI program. *Computational Science & Engineering*, vol. 5, nr. 1, pp. 36–45, January–March 1998.

[55] H. Hahn. *A Student's Guide to UNIX*. McGraw Hill, 1993.

[56] K. M. Heal, M. L. Hansen şi K. M. Rickard. *Maple V Learning Guide*. Springer-Verlag, 1996.

[57] R. T. Herbst. *Software design using FORTRAN77*. Prentice Hall, 1990.

[58] Kamran Husain şi Tim Parker. *Linux Unleashed*. Sams Publishing, Indianopolis, 1995.

[59] Daniel Ioan, Irina Munteanu, Bogdan Ionescu, Mihai Popescu, Radu Popa, Mihai Lăzărescu şi Gabriela Ciuprina. *Metode numerice în ingineria electrică*. MATRIX ROM, Bucureşti, România, 1998.

[60] M. Jalobeanu. *Internet, informare şi instruire*. Promedia Plus, 1995.

[61] B. W. Kernighan şi D. M. Ritchie. *The C Programming Language*. Prentice Hall, 1978.

[62] J. F. Kerrigan. *Migrating to Fortran90*. O'Reilly, 1993.

[63] Helmut Kopka şi Patrick Daly. *A Guide to LaTeX*. Addison–Wesley Publishing Co., 1993.

[64] Ed Krol. *Totul despre Internet*. RomSymData, 1995.

[65] J. R. Levine şi M. L. Young. *Unix pentru toţi*. Teora, 1996.

[66] G. McComb. *Web Programming Languages Sourcebook*. John Wiley, 1997.

[67] J. D. Murray şi W. Van Ryper. *Encyclopedia of Graphics File Formats*. O'Reilly, 1987.

[68] C. Musciano şi B. Kennedy. *HTML, The definitive Guide*. O'Reilly, 1997.

[69] North West Net. *The Internet Passport*. Prentice Hall, 1995.

[70] John Purcell, editor. *LINUX MAN – The Essential MAN Pages for Linux. Chap. III (Library Functions)*. Linux System Labs, 1995.

[71] A. Pusztai şi Gh. Ardelean. *LaTeX – Ghid de utilizare*. Editura Tehnică, 1994.

[72] V. Quercia şi T. O'Reilly. *XWindow System. User's Guide for X11 Release 5*. O'Reilly, 1993.

[73] K. Reichard şi E. F. Johnson. *Teach Yourself UNIX*. MIS Press, New York, 1992.

[74] J. R. Rice. *Migrating to Fortran90*. McGraw Hill, New York, 1983.

[75] M. A. Schoonover şi al. *GNU Emacs – UNIX Text Editing and Programming*. Addison–Wesley, 1992.

[76] A. Simpson. *Netscape Navigator Book*. Netscape Press, 1996.

[77] Bjarne Stroutstrup. *The C++ Programming Language*. Addison-Wesley Publishing Company, Wokingham, England, a doua ed., 1992.

[78] A. S. Tanenbaum. *Computer Networks*. Prentice Hall, 1996.

[79] Matt Welsh şi Lar Kaufman. *Running Linux*. O'Reilly & Associates, Sebastopol, CA, 1995.

[80] J. H. Wilkinson şi C. Reinsch. *Linear Algebra*. Springer Verlag, 1971.

[81] Stephen Wolfram. *The Mathematica Book*. Wolfram Media / Cambridge University Press, a treia ed., 1996.

[82] Timothy Van Zandt. `seminar.sty` – A LaTeX style for slides and notes. User's guide. `http://www.tex.ac.uk/tex-archive/macros/latex/contrib/ other/seminar/ doc/sem-guide.tex`, 1993.

Index

Simboluri

/ . . . 27, 28, 30
< . . . 32
< > . . . 208
> . . . 32
>> . . . 32
$ . . . 25, 26
% . . . 26
| . . . 32

A

`a2ps` . . . 32, 63
`abbrv`, stil bibliografic . . . 99, 103
`acm`, stil bibliografic . . . 100
Ajutor . . . *Vezi* Help
Alias . . . *Vezi* `elm` – alias
`alpha`, stil bibliografic . . . 99, 102
`apropos` . . . 34
Arhivare fişiere . . . 204
`article`, stil LaTeX . . . 71

B

`bash` . . . 26
`bg` . . . 33
Bibliografie . . . 96
  extensii nume fişiere . . . 100
  fişier `.bib` . . . 97
  includere în fişier `.tex` . . . 100
  introducerea referinţelor în LaTeX 99
  stil bibliografic . . . 99
    `abbrv` . . . 99, 103
    `acm` . . . 100
    `alpha` . . . 99, 102
    `ieee` . . . 100
    `ieeetran` . . . 100
    `plain` . . . 99, 101
    `plainrom` . . . 99
    `unsrt` . . . 99, 102
    `unsrtrom` . . . 99, 103
    utilitare . . . 96
BibTeX . . . *Vezi* Bibliografie
`bibtex` . . . 100
Bitmap . . . *Vezi* Figuri – bitmap, *Vezi* Figuri – bitmap
`book`, stil LaTeX . . . 71
Browser . . . 198, *Vezi* Internet – browser
BSD . . . 18

C

Căsută poştală . . . *Vezi* Poştă electronică
Căutări . . . 31
Căutare în Internet . . . *Vezi* Netscape
`cal` . . . 34
`calc` . . . 35
Calculator de buzunar *Vezi* `calc, xcalc`
Calcule matematice . . . 150
Captură de ecran/fereastră . . . *Vezi* `xv`
`cat` . . . 29
`Cc` . . . 57, 61
`cd` . . . 30
CGI . . . 198
`chmod` . . . 30
`clear` . . . 35
Client/server . . . 20, 193, 224
Comenzi Unix . . . 21–35
Comparări de fişiere . . . 31
Compilare . . . *Vezi* Programare – compilatoare
`compress` . . . 199, 204
Computational Science and . . . Engineering . . . *Vezi* CSE
Comunicaţie sub Unix . . . 193
  `compress, uncompress` . . . 204
  `ftp` . . . 203
  `gzip, gunzip` . . . 204
  `rlogin` . . . 204
  `ssh` . . . 204

- talk 201
- tar 204
- telnet 200
- uuencode, uudecode 204
- write 200

Configurare
- editorul vi 45
- elm 62
- LaTeX 104
- Scilab 151
- shell 25
- variabile Unix 25

Conversii de formate grafice ... 128, 135, 140, 142
Cota de spaţiu de disc 35
cp 29
CSE 9, 147
csh 26

D

date 34
Depanator (debugger) 176
Desene 121
- LaTeX 80
- reprezentare vectorială 122
- XFig 125
- XPaint 138

diff 31
Directoare 27
- speciale 27
  - curent 27
  - home 28
  - precedent 28
  - rădăcină 28

Documente 67
DOS → Unix 34
Drepturi asupra fişierelor 30
DVI 68
dvips 70

E

e-mail .... *Vezi* Poştă electronică
echo 34
Editare grafică .... *Vezi* Figuri
Editarea documentelor 39
Editarea textelor 39
Editoare de texte
- emacs 47
- vi 40

electronic mail ... *Vezi* Poştă electronică
elm 59, 199
- comenzi
  - citire mesaje 59
  - creare alias 61
  - ştergere/mutare mesaje 62
  - trimitere mesaje 60
- configurare 62

emacs 47
- comenzi
  - ştergere 49
  - încărcare fişier 50
  - înlocuire 50
  - acces la help 51
  - căutare 50
  - deplasare cursor 49
  - dezvoltare programe 51
  - ferestre 50
  - formatare 49
  - indentare 49
  - macrocomenzi 50
  - manipulare blocuri 50
  - mod dezvoltare programe 51
  - salvare fişier 50
  - Unix 51
- ieşire 47
- lansare 47
- mod LaTeX 51
- spelling 51
- structură ecran 47
- tutorial 51

Ethernet 195
Exemple de utilizare
- apel FORTRAN din C 179
- apel rutine externe din Scilab ... 164
- configurare LaTeX 105
- configurare vi 45
- gnuplot 136
- HTML 210
- index 117
- LaTeX 83–95
- make 109, 175

planşe de prezentare . . . 107
Scilab . . . 160
`tar` . . . 204
traducere Scilab → FORTRAN . 166
`exit` . . . 24
Extensii nume fişiere
.adi . . . 122
.aux . . . 83
.bbl . . . 100
.bib . . . 97, 99, 100, 104
.bin . . . 157
.blg . . . 100
.bmp . . . 123
.bst . . . 100, 104
.c, .cc, .f, .h, .o . . . 173
.dvi . . . 83
.eps . . . 123
.fig . . . 122, 128
.gif . . . 122
.glo . . . 83
.gnu . . . 136
.hpg . . . 122
.html . . . 207
.idx . . . 83
.ist . . . 118
.jpg . . . 123
.lof . . . 83
.log . . . 83
.lot . . . 83
.pbm . . . 123
.pcx . . . 123
.pdf . . . 123
.pgm . . . 123
.pk . . . 104
.pov . . . 123
.ppm . . . 123
.ps . . . 123
.sci . . . 157, 158
.sty . . . 104
.tex . . . 83, 104
.tif . . . 123
.toc . . . 83
.vrm . . . 123
.wmf . . . 123
.xbm . . . 123
.xpm . . . 123
fişiere grafice . . . 122–123

F

`f2c` . . . 170
Fast Ethernet . . . 195
Fereastră . . . *Vezi* XWindow
`fg` . . . 33
Fişiere . . . 27
afişare listă . . . 29
comparare . . . 31
copiere . . . 29
imprimare . . . 32
listare conţinut . . . 29
mutare . . . 29
nume . . . 27
redenumire . . . 29
schimbarea drepturilor . . . 30
ştergere . . . 29
Fişiere de comenzi
`gnuplot` . . . 136
Scilab . . . 158
Unix . . . *Vezi* Script
Figuri . . . 121
apel din meniuri . 239, 241, 242, 244, 245
bitmap . . . 122
incluse în HTML . . . 209
LaTeX . . . 72, 73, 80, 83
metafile . . . 122
PostScript . . 123, 125, 128, 135, 140, 142, 159, 160
XFig . . . 125, 159, 160
XPaint . . . 138
XV . . . 142
`file` . . . 205
`find` . . . 31
`finger` . . . 33, 199
Fire torsadate . . . *Vezi* UTP
Format fişier . *Vezi* Extensii nume fişiere
Formatarea documentelor . . . *Vezi* LaTeX
Formule matematice . . . *Vezi* LaTeX
FORTRAN 77 . . . 169, 172
Fortran 90 . . . 169
forward . . . 57
`fromdos` . . . 34

ftp 199, 203, 205

G

g++ 172
g77 172
gcc 172
gdb 176
ghostview 32, 70
GNU 19, 172
gnuplot 129
gopher 205
GPL 19
Grafice 121
  Gnuplot 129
  Scilab 159
grep 31
GUI 17
gunzip 204
gzip 199, 204

H

HDCL 195
Help . . 34, 45, 51, 59, 129, 134, 135, 139, 140, 142, 158
Hipermedia 198
hostname 21
HP-UX 18
HTML 205
  coduri de formatare 208–210
  structură 207
  tags 208–210

I

Ieşire în sistem 24
ieee, stil bibliografic 100
ieeetran, stil bibliografic 100
Imprimare . . . . . 32, *Vezi* lpr
\index 117
Indirectări 32
Internet 193, 195
  browser 205
  domenii naţionale 56
  domenii SUA 56
  domeniul .ro 198
  file 205
  ftp 205
  gopher 205
  HTML 205
  istoric 194
  URL 205
Interoperabilitate 170
Intrare în sistem 21
IP 195
IPv6 195
IRIX 18
Istoricul LMN 223

J

Java 198
JavaScript 198
Joe 40

K

kill 33
ksh 26

L

LAN . . . . . *Vezi* Local Area Network
Lansare/oprire comenzi Unix 33
LaTeX
  accente 81
  cadru 79
  caractere speciale 81
  comenzi 74
  configurare 104
  definire de noi comenzi 80
  despărţire în silabe 82
  diacritice pentru limba română . . . 81
  exemple de utilizare 83
  extensii nume fişiere 83
  formule matematice 86
  index 80
  macrodefiniţii 80
  makefile 109
  modul LR 78
  modul matematic 77
  modul paragraf 75
  moduri de lucru 75
  realizarea fişierului imprimabil . . . 70
  stiluri
    article 71
    book 71

letter 71
report 71
structură fişier 70
structura paginii 83
tipuri de comenzi 74
latex 70
latex2html 214
letter, stil LaTeX 71
Linux 19
LMN
adresă WWW 227
istoric 223
meniuri de comenzi 239
regulament 221
resurse hardware 230, 232
resurse software 237
Local Area Network 193
locate 31
lock 25
login 21
logout 24
lpq 33
lpr 32, 70
ls 29
Lucru cu directoare 30
Lucru cu ferestre X 21, 35, 130
Lucru cu fişiere 29
Lucrul cu ferestre X
.... *Vezi şi* XWindow
Lynx 205

M

Macro .... *Vezi* Scilab, LaTeX, emacs
Make
pentru compilare 174
pentru procesarea fişierelor .tex 109
makeidx 117
\makeindex 117
makeindex 117
MAN .. *Vezi* Metropolitan Area Network
man 34
MAPLE 149
Mathematica 147
MATLAB 148
Medii de programare 169
Meniu .... *Vezi* XWindow
Mesaje ... *Vezi* Poştă electronică, write
mesg 199
Metafile .... *Vezi* Figuri – metafile
Metropolitan Area Network 193
mkdir 30
Modelare
fizică 147
matematică 147
numerică 147
more 29
mv 29

N

Navigaţie în Internet ..... *Vezi* Netscape
NetLib 170
Netscape 205
Nucleul sistemului de operare 17

O

OOP 170
Open System Interconection ... *Vezi* OSI
OSI 195

P

Parolă .... *Vezi* password
passwd 25
password 21
Pearl 198
ping 199
plain, stil bibliografic 99, 101
plainrom, stil bibliografic 99
Planşe de prezentare 105
Poştă electronică 55
adresă 56
elm 59
reguli de stil 57
semnătură electronică 58
smileys 58
POSIX 19
PostScript 148, *Vezi* Figuri – PostScript
PPP 195
\printindex 117
Procesarea textelor 39
Programare
apel rutine FORTRAN
din C 179

din Scilab . . . 163
compilatoare . . . 172
depanare . . . 172, 176
editare programe în emacs . . . 51
make . . . 174
medii . . . 169
orientată pe obiecte . . . 170
în Scilab . . . 156
stilul profesional . . . 173
structurată . . . 170
traducere automată
FORTRAN → C . . . 170
Scilab → FORTRAN . . . 166
ps . . . 33
Pseudonim . . . *Vezi* alias
pwd . . . 30

Q

quota . . . 35

R

Reţea de calculatoare . . . 193
Referat de laborator . . . 111, 112
Regulament LMN . . . 221
report, stil LaTeX . . . 71
Reprezentare metafişier . . . *Vezi* Figuri – metafile
Reprezentare pixel *Vezi* Figuri – bitmap
Reprezentare vectorială . . *Vezi* Desene – reprezentare vectorială
Request For Comments . . . *Vezi* RFC
Resurse hardware LMN . . . 230, 232
Resurse software LMN . . . 237
RFC . . . 196
rlogin . . . 204
rm . . . 29
rmdir . . . 30
RoEduNet . . . 194, 198, 225

S

Salvare mesaje . . . *Vezi* Poştă electronică
SCICOS . . . 149
Scilab . . . 147
apel de rutine externe . . . 163
componente . . . 150
demo . . . 152
fişiere de comenzi . . . 158
formatul comenzilor . . . 151
funcţii predefinite . . . 157
grafice . . . 159
instrucţiuni . . . 157
programare . . . 156
structuri de date . . . 152
traducere automată . . .
în FORTRAN . . . 166
scilab . . . 150
Script . . . 26, 237
sdiff . . . 31
seminar . . . 105
SGML . . . 207
sh . . . 26
Shell Unix . . . 17
SLIP . . . 195
Smileys . . . 58
SMTP . . . 195
SNMP . . . 195
Solaris . . . 18
sort . . . 35
ssh . . . 24, 26, 204
Standard General . . .
Markup Language . . *Vezi* SGML
Structura de directoare . . . 26
Subject . . . 57
Sun OS . . . 18
SVR4 . . . 18

T

Tabele . . . 72, 73, 80
talk . . . 199, 201
tar . . . 199, 204
TCP . . . 195
Telnet . . . 195
telnet . . . 199, 200
Teste de verificare . . . 216–220
TeX . . . 68, 148
Text . . . 39
todos . . . 34
Toolbox . . . 11, 23
Translatoare
ASCII → PostScript . . . 32, 63
FORTRAN → C . . . 170
LaTeX → HTML . . . 214

U

Ultrix 18
`umask` 31
`uname` 21
`uncompress` 204
Uniform Resource Locator ... *Vezi* URL
UNIX 11, 18
Unix 11, 18–20
Unshielded Twisted Pair ..... *Vezi* UTP
`unsrt`, stil bibliografic 99, 102
`unsrtrom`, stil bibliografic 99, 103
URL 205
Usenet 195
username 21
UTP 194
`uudecode` 199, 204
`uuencode` 199, 204

V

Vectorial .... *Vezi* Desene
`vi` 40
  comenzi
    înlocuiri 44
    căutări 44
    deplasare cursor 42
    editare 43
    ieşire 44
    salvare 44
  configurare 45
  intrare în mod text 42
  moduri de lucru 41
    comandă 41
    text 41
Vizualizare ..... *Vezi* `ghostview, xdvi`

W

`w` 33
WAN .... *Vezi* Wide Area Network
`wc` 35
Web .... *Vezi* WWW
Web page 207
Web site 207
`whatis` 34
`who` 33
`whoami` 33
Wide Area Network 193
World Wide Web .... *Vezi* WWW
`write` 199, 200
WWW 198, 205
WYSIWYG 40, 67, 198

X

X11 20
`xcalc` 35
`xclock` 35
`xdvi` 70
Xedit 40
Xenix 18
`xfig` 125
`xlock` 25
`xpaint` 138
`xterm` 35
`xv` 142
XWindow 20, 21, 130
`xxgdb` 176